Aligning Packed Dependency Trees: A Theory of Composition for Distributional Semantics

David Weir*
University of Sussex

Julie Weeds*
University of Sussex

Jeremy Reffin*
University of Sussex

Thomas Kober*
University of Sussex

We present a new framework for compositional distributional semantics in which the distributional contexts of lexemes are expressed in terms of anchored packed dependency trees. We show that these structures have the potential to capture the full sentential contexts of a lexeme and provide a uniform basis for the composition of distributional knowledge in a way that captures both mutual disambiguation and generalization.

1. Introduction

This article addresses a central unresolved issue in distributional semantics: how to model semantic composition. Although there has recently been considerable interest in this problem, it remains unclear what distributional composition actually means. Our view is that distributional composition is a matter of contextualizing the lexemes being composed. This goes well beyond traditional word sense disambiguation, where each lexeme is assigned one of a fixed number of senses. Our proposal is that composition involves deriving a fine-grained characterization of the distributional meaning of each lexeme in the phrase, where the meaning that is associated with each lexeme is bespoke to that particular context.

Distributional composition is, therefore, a matter of integrating the meaning of each of the lexemes in the phrase. To achieve this we need a structure within which all of the lexemes' semantics can be overlaid. Once this is done, the lexemes can collectively agree on the semantics of the phrase, and in so doing, determine the semantics that they have

* Department of Informatics, University of Sussex. Falmer, Brighton BN1 9QH, UK. E-mail: {d.j.weir, j.e.weeds, j.p.reffin, t.kober}@sussex.ac.uk.

Submission received: 10 April 2015; revised version received: 17 April 2016; accepted for publication: 20 April 2016.

doi:10.1162/COLI_a_00265

© 2017 Association for Computational Linguistics

in the context of that phrase. Our process of composition thus creates a single structure that encodes contextualized representations of every lexeme in the phrase.

The (uncontextualized) distributional knowledge of a lexeme is typically formed by aggregating distributional features across *all* uses of the lexeme found within the corpus, where distributional features arise from co-occurrences found in the corpus. The distributional features of a lexeme are associated with weights that encode the strength of that feature. Contextualization involves inferring adjustments to these weights to reflect the context in which the lexeme is being used. The weights of distributional features that don't fit the context are reduced, while the weight of those features that are compatible with the context can be boosted.

As an example, consider how we contextualize the distributional features of the word *wooden* in the context of the phrase *wooden floor*. The uncontextualized representation of *wooden* presumably includes distributional features associated with different uses, for example, *The director fired the wooden actor* and *I sat on the wooden chair*. So, although we may have observed in a corpus that it is plausible for the adjective *wooden* to modify *floor*, *table*, *toy*, *actor*, and *voice*, in the specific context of the phrase *wooden floor*, we need to find a way to down-weight the distributional features of being something that can modify *actor* and *voice*, while up-weighting the distributional features of being something that can modify *table* and *toy*.

In this example we considered so-called first-order distributional features; these involve a single dependency relation (e.g., an adjective modifying a noun). Similar inferences can also be made with respect to distributional features that involve higher-order grammatical dependencies.[1] For example, suppose that we have observed that a noun that *wooden* modifies (e.g., *actor*) can be the direct object of the verb *fired*, as in *The director fired the wooden actor*. We want this distributional feature of *wooden* to be down-weighted in the distributional representation of *wooden* in the context of *wooden table*, since things made of wood do not typically lose their job.

In addition to specializing the distributional representation of *wood* to reflect the context *wooden floor*, the distributional representation of *floor* should also be refined, down-weighting distributional features arising in contexts such as *Prices fell through the floor*, while up-weighting distributional features arising in contexts such as *I polished the concrete floor*.

In our example, some of the distributional features of *wooden*—in particular, those to do with the noun that this sense of *wooden* could modify—are **internal** to the phrase *wooden floor* in the sense that they are alternatives to one of the words in the phrase. Although it is specifically a *floor* that is *wooden*, our proposal is that the contextualized representation of *wooden* should recognize that it is plausible that nouns such as *chair* and *toy* could be modified by the particular sense of *wooden* that is being used. The remaining distributional features are **external** to the phrase. For example, the verb *mop* could be an external feature, because things that can be modified by *wooden* can be the direct object of *mop*. The external features of *wooden* and *floor* with respect to the phrase *wooden floor* provide something akin to the traditional interpretation of the distributional semantics of the phrase, namely, a representation of those (external) contexts in which this phrase can occur.

Although internal features are, in a sense, inconsistent with the specific semantics of the phrase, they provide a way to embellish the characterization of the distributional

1 Given some dependency tree, a k-th order dependency holds between two lexemes (nodes) in the tree when the path between the two lexemes has length k.

meaning of the lexemes in the phrase. Recall that our goal is to infer a rich and fine-grained representation of the contextualized distributional meaning of each of the lexemes in the phrase.

Having introduced the proposal that distributional composition should be viewed as a matter of contextualization, the question arises as to how to realize this conception. Because each lexeme in the phrase needs to be able to contribute to the contextualization of the other lexemes in the phrase, we need to be able to align what we know about each of the lexeme's distributional features so that this can be achieved. The problem is that the uncontextualized distributional knowledge associated with the different lexemes in the phrase take a different perspective on the feature space. To overcome this we need to: (a) provide a way of structuring the distributional feature space, which we do by typing distributional features with dependency paths; and (b) find a way to systematically modify the perspective that each lexeme has on this structured feature space in such a way that they are all aligned with one another.

Following Baroni and Lenci (2010), we use typed dependency relations as the bases for our distributional features, and following Padó and Lapata (2007), we include higher-order dependency relations in this space. However, in contrast to previous proposals, the higher-order dependency relations provides structure to the space that is crucial to our definition of composition. Each co-occurrence associated with a lexeme such as *wooden* is typed by the path in the dependency tree that connects the lexeme *wooden* with the co-occurring lexeme (e.g., *fired*). This allows us to encode a lexeme's distributional knowledge with a hierarchical structure that we call an Anchored Packed Dependency Tree (APT). As we show, this data structure provides a way for us to align the distributional knowledge of the lexemes that are being composed in such a way that the inferences needed to achieve contextualization can be implemented.

2. The Distributional Lexicon

In this section, we begin the formalization of our proposal by describing the distributional lexicon: a collection of entries that characterize the distributional semantics of lexemes. Table 1 provides a summary of the notation that we are using.

Let V be a finite alphabet of lexemes,[2] where each lexeme is assumed to incorporate a part-of-speech tag; let R be a finite alphabet of grammatical dependency relations; and let $T_{V,R}$ be the set of dependency trees where every node is labeled with a member of V, and every directed edge is labeled with an element of R. Figure 1 shows eight examples of dependency trees.

2.1 Typed Co-occurrences

When two lexemes w and w' co-occur in a dependency tree[3] in $t \in T_{V,R}$, we represent this co-occurrence as a triple $\langle w, \tau, w' \rangle$ where τ is a string that encodes the **co-occurrence type** of this co-occurrence, capturing the syntactic relationship that holds between these

2 There is no reason why lexemes could not include multi-word phrases tagged with an appropriate part of speech.

3 In order to avoid over-complicating our presentation, when possible, we do not distinguish between a node in a dependency tree and the lexeme that appears at that node.

Table 1
Summary of notation.

Notation	Description
V	a finite set of lexemes
w	a lexeme
R	a finite set of dependency tree edge labels
r	an element of R
$\bar{R}$	a finite set of inverse dependency tree edge labels
$\bar{r}$	an element of $\bar{R}$
x	an element of $R \cup \bar{R}$
$T_{V,R}$	the dependency trees over lexemes V and dependencies R
t	a dependency tree
τ	a co-occurrence type (path)
τ^{-1}	the inverse (reverse) of path τ
$\langle w, \tau, w' \rangle$	the co-occurrence of w with w' with co-occurrence type τ
C	a corpus of (observed) dependency trees
$\downarrow(\tau)$	the co-occurrence type produced by reducing τ
$\#(\langle w, \tau, w' \rangle, t)$	number of occurrences of $\langle w, \tau, w' \rangle$ in t
$\#\langle w, \tau, w' \rangle$	number of occurrences of $\langle w, \tau, w' \rangle$ in the corpus
$\lVert w \rVert$	the (uncontextualized) APT for w
$\mathbf{A}$	an APT
$\lVert w \rVert(\tau, w')$	the weight for w' in $\lVert w \rVert$ at node for co-occurrence type τ
$\lVert w \rVert(\tau)$	the node (weighted lexeme multiset) in $\lVert w \rVert$ for co-occurrence type τ
FEATS	the set of all distributional features arising in C
$\langle \tau, w \rangle$	a distributional feature in vector space
$\mathrm{W}(w, \langle \tau, w' \rangle)$	the weight of the distributional feature $\langle \tau, w' \rangle$ of lexeme w
$\overrightarrow{\lVert w \rVert}$	the vector representation of the APT $\lVert w \rVert$
$\mathrm{SIM}(\lVert w_1 \rVert, \lVert w_2 \rVert)$	the distributional similarity of $\lVert w_1 \rVert$ and $\lVert w_2 \rVert$
$\lVert w \rVert^{\delta}$	the APT $\lVert w \rVert$ that has been offset by δ
$\lVert t \rVert$	the composed APT for the tree t
$\lVert w; t \rVert$	the APT for w when contextualized by t
$\bigsqcup \{ \mathbf{A}_1, \ldots, \mathbf{A}_n \}$	the result of merging aligned APTs in $\{ \mathbf{A}_1, \ldots, \mathbf{A}_n \}$

occurrences of the two lexemes. In particular, τ encodes the sequence of dependencies that lie along the path in t between the occurrences of w and w' in t. In general, a path from w to w' in t initially travels up towards the root of t (against the directionality of the dependency edges) until an ancestor of w' is reached. It then travels down the tree to w' (following the directionality of the dependencies). The string τ must, therefore, not only encode the sequence of dependency relations appearing along the path, but also whether each edge is traversed in a forward or backward direction. In particular, given the path $\langle v_0, \ldots, v_k \rangle$ in t, where $k > 0$, w labels v_0, and w' labels v_k, the string $\tau = x_1 \ldots x_k$ encodes the co-occurrence type associated with this path as follows:

- If the edge connecting v_{i-1} and v_i runs from v_{i-1} to v_i and is labeled by r, then $x_i = r$.
- If the edge connecting v_{i-1} and v_i runs from v_i to v_{i-1} and is labeled by r, then $x_i = \bar{r}$.

Hence, co-occurrence types are strings in $\overline{R}^* R^*$, where $\overline{R} = \{ \bar{r} \mid r \in R \}$.

It is useful to be able to refer to the **order** of a co-occurrence type, where this simply refers to the length of the dependency path. It is also convenient to be able to refer to the inverse of a co-occurrence type. This can be thought of as the same path, but traversed in the reverse direction. To be precise, given the co-occurrence type $\tau = x_1 \cdot \ldots \cdot x_n$ where each $x_i \in R \cup \overline{R}$ for $1 \leq i \leq n$, the inverse of τ, denoted τ^{-1}, is the path ${x_n}^{-1} \cdot \ldots \cdot {x_1}^{-1}$ where $r^{-1} = \bar{r}$ and $\bar{r}^{-1} = r$ for $r \in R$. For example, the inverse of $\overline{\text{AMOD}} \cdot \text{DOBJ} \cdot \text{NSUBJ}$ is $\overline{\text{NSUBJ}} \cdot \overline{\text{DOBJ}} \cdot \text{AMOD}$.

The following typed co-occurrences for the lexeme *white*/JJ arise in the tree shown in Figure 1(a).

$\langle$*white*/JJ, $\overline{\text{AMOD}} \cdot \overline{\text{DOBJ}} \cdot \text{NSUBJ}$, *we*/PRP$\rangle$

$\langle$*white*/JJ, $\overline{\text{AMOD}} \cdot \overline{\text{DOBJ}}$, *bought*/VBD$\rangle$

$\langle$*white*/JJ, $\overline{\text{AMOD}} \cdot \text{DET}$, *the*/DT$\rangle$

$\langle$*white*/JJ, $\overline{\text{AMOD}} \cdot \text{AMOD} \cdot \text{ADVMOD}$, *slightly*/RB$\rangle$

$\langle$*white*/JJ, $\overline{\text{AMOD}} \cdot \text{AMOD}$, *fizzy*/JJ$\rangle$

$\langle$*white*/JJ, $\overline{\text{AMOD}} \cdot \text{AMOD}$, *dry*/JJ$\rangle$

$\langle$*white*/JJ, ϵ, *white*/JJ$\rangle$

$\langle$*white*/JJ, $\overline{\text{AMOD}}$, *wine*/NN$\rangle$

Notice that we have included the co-occurrence $\langle$*white*/JJ, ϵ, *white*/JJ$\rangle$. This gives a uniformity to our typing system that simplifies the formulation of distributional composition in Section 4, and leads to the need for a refinement to our co-occurrence type encodings. Because we permit paths that traverse both forwards and backwards along the same dependency—for example, in the co-occurrence $\langle$*white*/JJ, $\overline{\text{AMOD}} \cdot \text{AMOD}$, *dry*/JJ$\rangle$—it is logical to consider $\langle$*white*/JJ, $\overline{\text{AMOD}} \cdot \overline{\text{DOBJ}} \cdot \text{DOBJ} \cdot \text{AMOD}$, *dry*/JJ$\rangle$ a valid co-occurrence. However, in line with our decision to include $\langle$*white*/JJ, ϵ, *white*/JJ$\rangle$ rather than $\langle$*white*/JJ, $\overline{\text{AMOD}} \cdot \text{AMOD}$, *white*/JJ$\rangle$, all co-occurrence types are canonicalized through a dependency cancellation process in which adjacent, complementary dependencies are cancelled out. In particular, all occurrences within the string of either $r\bar{r}$ or $\bar{r}r$ for $r \in R$ are replaced with ϵ, and this process is repeated until no further reductions are possible.

The reduced co-occurrence type produced from τ is denoted $\downarrow(\tau)$, and defined as follows:

$$\downarrow(\tau) = \begin{cases} \downarrow(\tau_1 \tau_2) & \text{if } \tau = \tau_1\, r\, \bar{r}\, \tau_2 \text{ or } \tau = \tau_1\, \bar{r}\, r\, \tau_2 \text{ for some } r \in R \\ \tau & \text{otherwise} \end{cases} \tag{1}$$

For the remainder of this article, we only consider reduced co-occurrence types when associating a type with a co-occurrence.

Given a tree $t \in T_{V,R}$, lexemes w and w', and reduced co-occurrence type τ, the number of times that the co-occurrence $\langle w, \tau, w' \rangle$ occurs in t is denoted $\#(\langle w, \tau, w' \rangle, t)$, and, given some corpus C of dependency trees, the sum of all $\#(\langle w, \tau, w' \rangle, t)$ across all $t \in C$ is denoted $\#\langle w, \tau, w' \rangle$. Note that in order to simplify our notation, the dependence on the corpus C is not expressed in our notation.

It is common to use alternatives to raw counts in order to capture the strength of each distributional feature. A variety of alternatives are considered during the experimental work presented in Section 5. Among the options we have considered are probabilities and various versions of positive pointwise mutual information. Although, in practice, the precise method for weighting features is of practical importance, it is not an intrinsic part of the theory that this article is introducing. In the following exposition

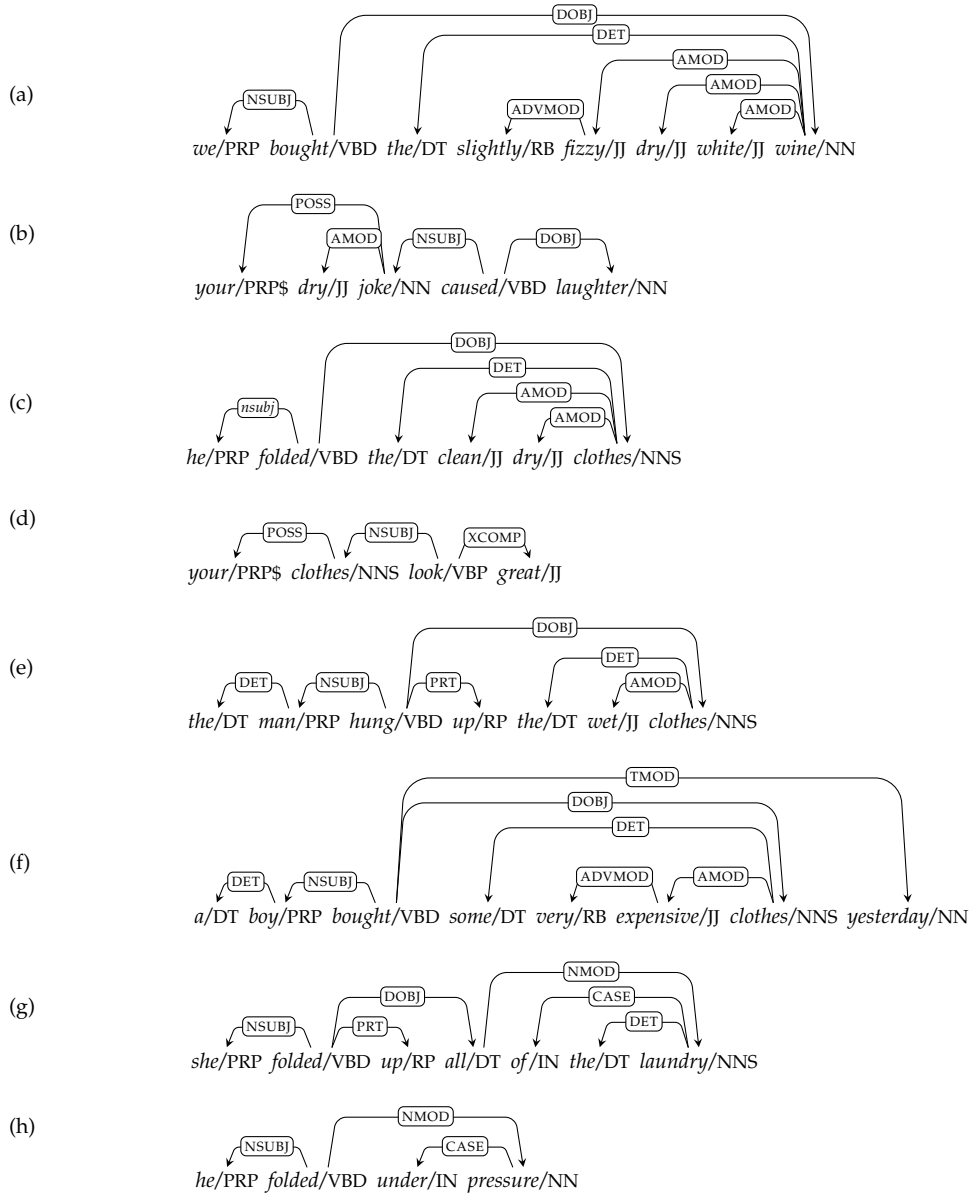

Figure 1
A small corpus of dependency trees.

we denote the weight of the distributional feature $\langle \tau, w' \rangle$ of the lexeme w with the expression $\mathrm{W}(w, \langle \tau, w' \rangle)$.

2.2 Anchored Packed Trees

Given a dependency tree corpus $C \subset T_{V,R}$ and a lexeme $w \in V$, we are interested in capturing the aggregation of all distributional contexts of w in C within a single structure. We achieve this with what we call an **Anchored Packed Dependency Tree** (APT). APTs

are central to the proposals in this article: Not only can they be used to encode the aggregate of all distributional features of a lexeme over a corpus of dependency trees, but they can also be used to express the distributional features of a lexeme that has been contextualized within some dependency tree (see Section 4).

The APT for w given C is denoted $\|w\|$, and referred to as the **elementary APT** for w. In the following discussion, we describe a tree-based interpretation of $\|w\|$, but in the first instance we define it as a mapping from pairs (τ, w') where $\tau \in \overline{R}^* R^*$ and $w' \in V$, such that $\|w\|(\tau, w')$ gives the weight of the typed co-occurrence $\langle w, \tau, w' \rangle$ in the corpus C. It is nothing more than those components of the weight function that specify the weights of distributional features of w. In other words, for each $\tau \in \overline{R}^* R^*$ and $w' \in V$:

$$\|w\|(\tau, w') = \mathrm{W}(w, \langle \tau, w' \rangle) \tag{2}$$

The restriction of $\|w\|$ to co-occurrence types that are at most order k is referred to as a k-th order APT. The **distributional lexicon** derived from a corpus C is a collection of lexical entries where the entry for the lexeme w is the elementary APT $\|w\|$.

Formulating APTs as functions simplifies the definitions that appear herein. However, because an APT encodes co-occurrences that are aggregated over a set of dependency trees, they can also be interpreted as having a tree structure. In our tree-based interpretation of APTs, nodes are associated with weighted multisets of lexemes. In particular, $\|w\|(\tau)$ is thought of as a node that is associated with the weighted lexeme multiset in which the weight of w' in the multiset is $\|w\|(\tau, w')$. We refer to the node $\|w\|(\epsilon)$ as the **anchor** of the APT $\|w\|$.

Figure 2 shows three elementary APTs that can be produced from the corpus shown in Figure 1. On the far left we give the letter corresponding to the sentence in Figure 1 that generated the typed co-occurrences. Each column corresponds to one node in the APT, giving the multiset of lexemes at that node. Weights are not shown, and only non-empty nodes are displayed.

It is worth dwelling on the contents of the anchor node of the top APT in Figure 2, which is the elementary APT for *dry*/JJ. The weighted multiset at the anchor node is denoted $\|w\|(\epsilon)$. The lexeme *dry*/JJ occurs three times, and the weight $\|w\|(\epsilon, \textit{dry}/\text{JJ})$ reflects this count. Three other lexemes also occur at this same node: *fizzy*/JJ, *white*/JJ, and *clean*/JJ. These lexemes arose from the following co-occurrences in trees in Figure 1: $\langle \textit{dry}/\text{JJ}, \overline{\text{AMOD}} \cdot \text{AMOD}, \textit{fizzy}/\text{JJ} \rangle$, $\langle \textit{dry}/\text{JJ}, \overline{\text{AMOD}} \cdot \text{AMOD}, \textit{white}/\text{JJ} \rangle$, and $\langle \textit{dry}/\text{JJ}, \overline{\text{AMOD}} \cdot \text{AMOD}, \textit{clean}/\text{JJ} \rangle$, all of which involve the co-occurrence type $\overline{\text{AMOD}} \cdot \text{AMOD}$. These lexemes appear in the multiset $\|w\|(\epsilon)$ because $\downarrow(\overline{\text{AMOD}} \cdot \text{AMOD}) = \epsilon$.

3. APT Similarity

One of the most fundamental aspects of any treatment of distributional semantics is that it supports a way of measuring distributional similarity. In this section, we describe a straightforward way in which the similarity of two APTs can be measured through a mapping from APTs to vectors.

First, define the set of distributional features

$$\text{FEATS} = \{ \langle \tau, w' \rangle \mid w' \in V, \tau \in \overline{R}^* R^* \text{ and } \mathrm{W}(w, \langle \tau, w' \rangle) > 0 \text{ for some } w \in V \} \tag{3}$$

The vector space that we use to encode APTs includes one dimension for each element of FEATS, and we use the pair $\langle \tau, w \rangle$ to refer to its corresponding dimension.

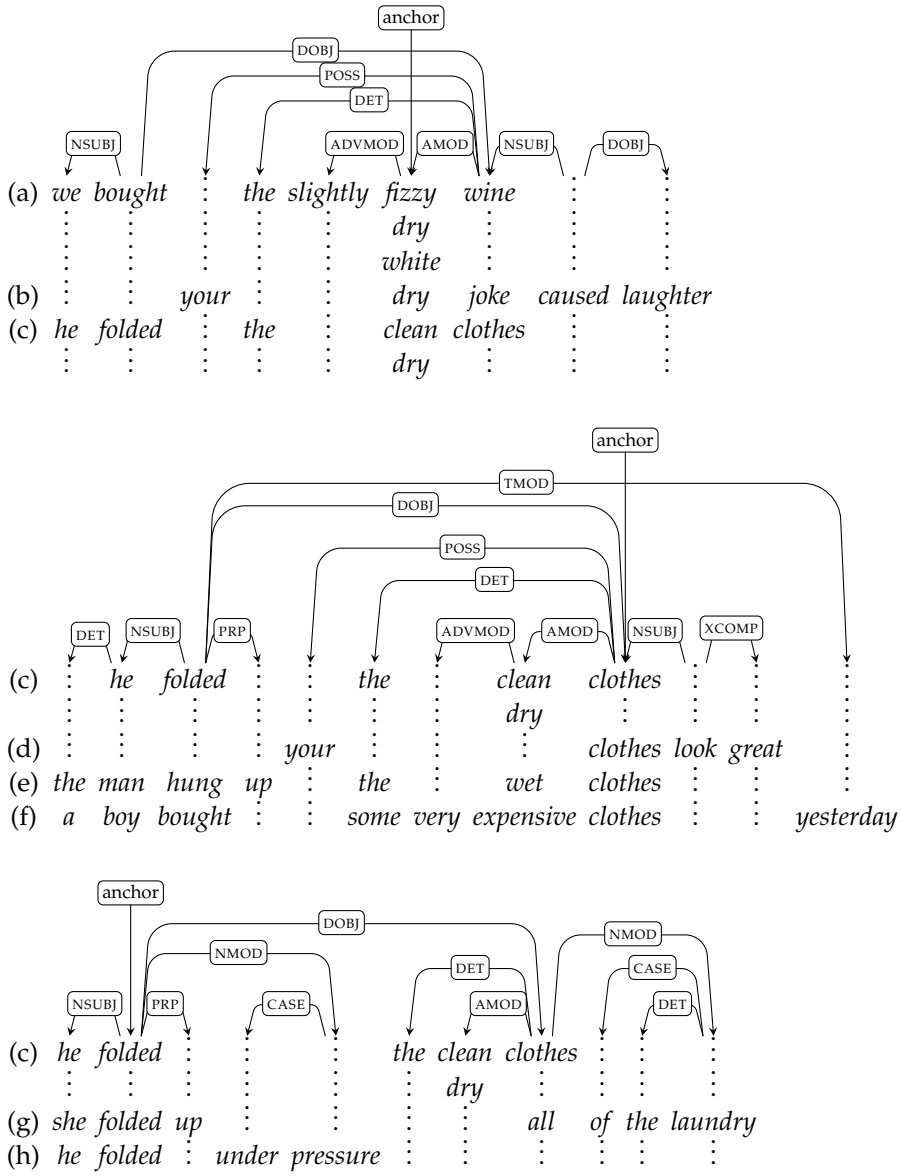

Figure 2
The distributional lexicon produced from the trees in Figure 1 with the elementary APT for *dry*/JJ at the top, the elementary APT for *clothes*/NNS in the middle, and the elementary APT for *folded*/VBD at the bottom. Part of speech tags and weights have been omitted.

Given an APT $\mathbf{A}$, we denote the vectorized representation of $\mathbf{A}$ with $\overrightarrow{\mathbf{A}}$, and the value that the vector $\overrightarrow{\mathbf{A}}$ has on dimension $\langle \tau, w' \rangle$ is denoted $\overrightarrow{\mathbf{A}}\left[\langle \tau, w' \rangle\right]$. For each $\langle \tau, w' \rangle \in$ FEATS:

$$\overrightarrow{\|w\|}\left[\langle \tau, w' \rangle\right] = \phi(\tau, w)\, \mathrm{W}(w, \langle \tau, w' \rangle) \tag{4}$$

where $\phi(\tau, w)$ is a path-weighting function that is intended to reflect the fact that not all of the distributional features are equally important in determining the distributional similarity of two APTs. Generally speaking, syntactically distant co-occurrences provide a weaker characterization of the semantics of a lexeme than co-occurrences that are syntactically closer. By multiplying each $\mathrm{W}(w, \langle \tau, w' \rangle)$ by $\phi(\tau, w)$, we are able to capture this, given a suitable instantiation of $\phi(\tau, w)$.

One option for $\phi(\tau, w)$ is to use $p(\tau \mid w)$—that is, the probability that when randomly selecting one of the co-occurrences $\langle w, \tau', w' \rangle$, where w' can be any lexeme in V, τ' is the co-occurrence type τ. We can estimate these path probabilities from the co-occurrence counts in C as follows:

$$p(\tau \mid w) = \frac{\#\langle w, \tau, * \rangle}{\#\langle w, *, * \rangle} \tag{5}$$

where

$$\begin{aligned} \#\langle w, \tau, * \rangle &= \textstyle\sum_{w' \in V} \#\langle w, \tau, w' \rangle \\ \#\langle w, *, * \rangle &= \textstyle\sum_{w' \in V} \sum_{\tau \in \bar{R}^* R^*} \#\langle w, \tau, w' \rangle \end{aligned}$$

$p(\tau \mid w)$ typically falls off rapidly as a function of the length of τ as desired.

The similarity of two APTs, $\mathbf{A}_1$ and $\mathbf{A}_2$, which we denote $\mathrm{SIM}(\mathbf{A}_1, \mathbf{A}_2)$, can be measured in terms of the similarity of vectors $\overrightarrow{\mathbf{A}_1}$ and $\overrightarrow{\mathbf{A}_2}$. The similarity of vectors can be measured in a variety of ways (Lin 1998; Lee 1999; Weeds and Weir 2005; Curran 2004). One popular option involves the use of the cosine measure:

$$\mathrm{SIM}(\mathbf{A}_1, \mathbf{A}_2) = \cos(\overrightarrow{\mathbf{A}_1}, \overrightarrow{\mathbf{A}_2}) \tag{6}$$

It is common to apply cosine to vectors containing positive pointwise mutual information (PPMI) values. If the weights used in the APTs are counts or probabilities, then they can be transformed into PPMI values at this point.

As a consequence of the fact that the different co-occurrence types of the co-occurrences associated with a lexeme are being differentiated, vectorized APTs are much sparser than traditional vector representations used to model distributional semantics. This can be mitigated in various ways, including:

- reducing the granularity of the dependency relations and/or the part-of-speech tag set
- applying various normalizations of lexemes such as case normalization, lemmatization, or stemming
- disregarding all distributional features involving co-occurrence types over a certain length
- applying some form of distributional smoothing, where distributional features of a lexeme are inferred based on the features of distributionally similar lexemes

4. Distributional Composition

In this section we turn to the central topic of the article, namely, distributional composition. We begin with an informal explanation of our approach, and then present a more precise formalization.

4.1 Discussion of Approach

Our starting point is the observation that although we have shown that all of the elementary APTs in the distributional lexicon can be placed in the same vector space (see Section 3), there is an important sense in which APTs for different parts of speech are not comparable. For example, many of the dimensions that make sense for verbs, such as those involving a co-occurrence type that begins with DOBJ or NSUBJ, do not make sense for a noun. However, as we now explain, the co-occurrence type structure present in an APT allows us to address this, making way for our definition of distributional composition.

Consider the APT for the lexeme *dry*/JJ shown at the top of Figure 2. The anchor of this APT is the node at which the lexeme *dry*/JJ appears. We can, however, take a different perspective on this APT—for example, one in which the anchor is the node at which the lexemes *bought*/VBD and *folded*/VBD appear. This APT is shown at the top of Figure 3. Adjusting the position of the anchor is significant because the starting point of the paths given by the co-occurrence types change. For example, when the APT shown at the top of Figure 3 is applied to the co-occurrence type $\overline{\text{NSUBJ}}$, we reach the node at which the lexemes *we*/PRP and *he*/PRP appear. Thus, this APT can be seen as a characterization of the distributional properties of the verbs that nouns that *dry*/JJ modifies can take as their direct object. In fact, it looks rather like the elementary APT for some verb. The lower tree in Figure 3 shows the elementary APT for *clothes*/NNS (the center APT shown in Figure 2), where the anchor has been moved to the node at which the lexemes *folded*/VBD, *hung*/VBD, and *bought*/VBD appear.

Notice that in both of the APTs shown in Figure 3, parts of the tree are shown in faded text. These are nodes and edges that are removed from the APT as a result of the change in anchor placement. The elementary tree for *dry*/JJ shown in Figure 2 reflects the fact that at least some of the nouns that *dry*/JJ modifies can be the direct object of a verb, or the subject of a verb. When we move the anchor, as shown at the top of Figure 3, we resolve this ambiguity to the case where the noun being modified is a direct object. The incompatible parts of the APT are removed. This corresponds to restricting the co-occurrence types of composed APTs to those that belong to the set $\overline{R}^*R^*$, just as was the case for elementary APTs. For example, note that in the upper APT of Figure 3, neither the path $\text{DOBJ} \cdot \overline{\text{NSUBJ}}$ from the node labeled with *bought*/VBD and *folded*/VBD to the node labeled *caused*/VBD, nor the path $\text{DOBJ} \cdot \overline{\text{SUBJ}} \cdot \overline{\text{DOBJ}}$ from the node labeled with *bought*/VBD and *folded*/VBD to the node labeled *laughter*/NN, are in $\overline{R}^*R^*$.

Given a sufficiently rich elementary APT for *dry*/JJ, those verbs that have nouns that *dry*/JJ can plausibly modify as direct objects have elementary APTs that are in some sense "compatible" with the APT produced by shifting the anchor node as illustrated at the top of Figure 3. An example is the APT for *folded*/VBD shown at the bottom of Figure 2. Loosely speaking, this means that, when applied to the same co-occurrence type, the APT in Figure 3 and the APT at the bottom of Figure 2 are generally expected to give sets of lexemes with related elements.

By moving the anchors of the APT for *dry*/JJ and *clothes*/NNS as in Figure 3, we have, in effect, aligned all of the nodes of the APTs for *dry*/JJ and *clothes*/NN with the

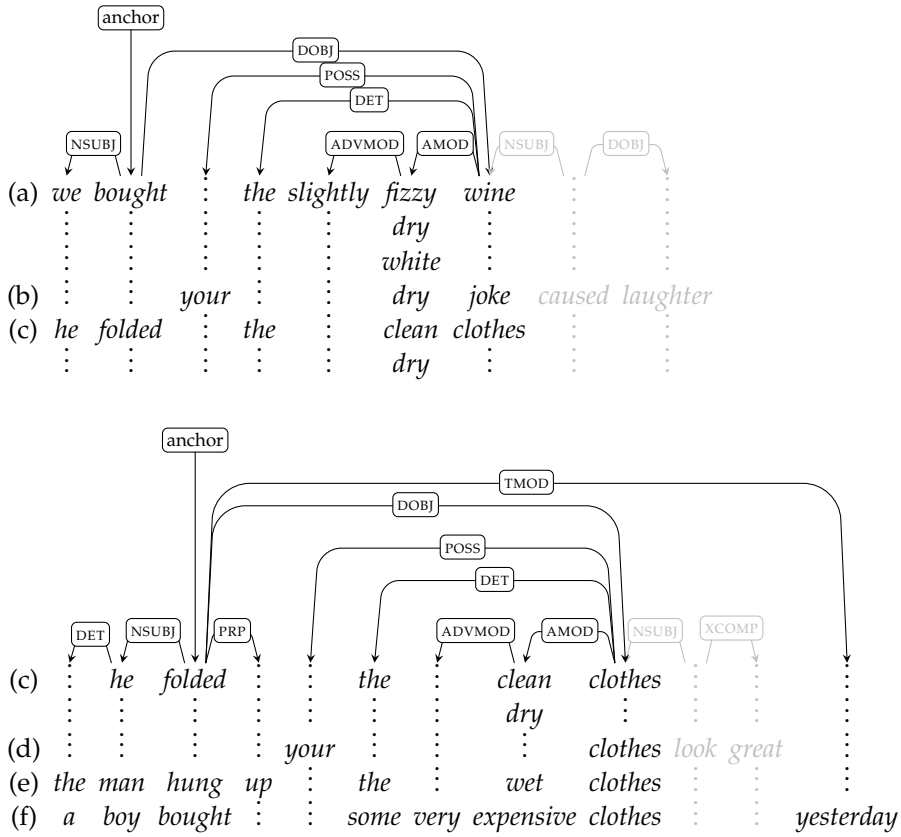

Figure 3
The elementary APTs for *dry*/JJ and *clothes*/NNS with anchors offset.

nodes they correspond to in the APT for *folded*/VBD. Not only does this make it possible, in principle at least, to establish whether or not the composition of *dry*/JJ, *clothes*/NNS, and *folded*/VBD is plausible, but it provides the basis for the contextualization of APTs, as we now explain.

Recall that elementary APTs are produced by aggregating contexts taken from all of the occurrences of the lexeme in a corpus. As described in the Introduction, we need a way to contextualize aggregated APTs in order to produce a fine-grained characterization of the distributional semantics of the lexeme in context. There are two distinct aspects to the contextualization of APTs, both of which can be captured through APT composition: **co-occurrence filtering**—the down-weighting of co-occurrences that are not compatible with the way the lexeme is being used in its current context; and **co-occurrence embellishment**—the up-weighting of compatible co-occurrences that appear in the APTs for the lexemes with which it is being composed.

Both co-occurrence filtering and co-occurrence embellishment can be achieved through APT composition. The process of composing the elementary APTs for the lexemes that appear in a phrase involves two distinct steps. First, the elementary APTs for each of the lexemes being composed are aligned in a way that is determined by the dependency tree for the phrase. The result of this alignment of the elementary APTs is that each node in one of the APTs is matched up with (at most) one of the nodes in

each of the other APTs. The second step of this process involves merging nodes that have been matched up with one another in order to produce the resulting composed APT that represents the distributional semantics of the dependency tree. It is during this second step that we are in a position to determine those co-occurrences that are compatible across the nodes that have been matched up.

Figure 4 illustrates the composition of APTs on the basis of a dependency tree shown in the upper center of the figure. In the lower right, the figure shows the full APT that results from merging the six aligned APTs, one for each of the lexemes in the dependency tree. Each node in the dependency tree is labeled with a lexeme, and around the dependency tree we show the elementary APTs for each lexeme. The six elementary APTs are aligned on the basis of the position of their lexeme in the dependency tree. Note that the tree shown in gray within the APT is structurally identical to the dependency tree in the upper center of the figure. The nodes of the dependency tree are labeled with single lexemes, whereas each node of the APT is labeled by a weighted lexeme multiset. The lexeme labeling a node in the dependency tree is one of the lexemes found in the weighted lexeme multiset associated with the corresponding node within the APT. We refer to the nodes in the composed APT that come from nodes in the dependency tree (the gray nodes) as the **internal context**, and the remaining nodes as the **external context**.

As we have seen, the alignment of APTs can be achieved by adjusting the location of the anchor. The specific adjustments to the anchor locations are determined by the dependency tree for the phrase. For example, Figure 5 shows a dependency analysis of

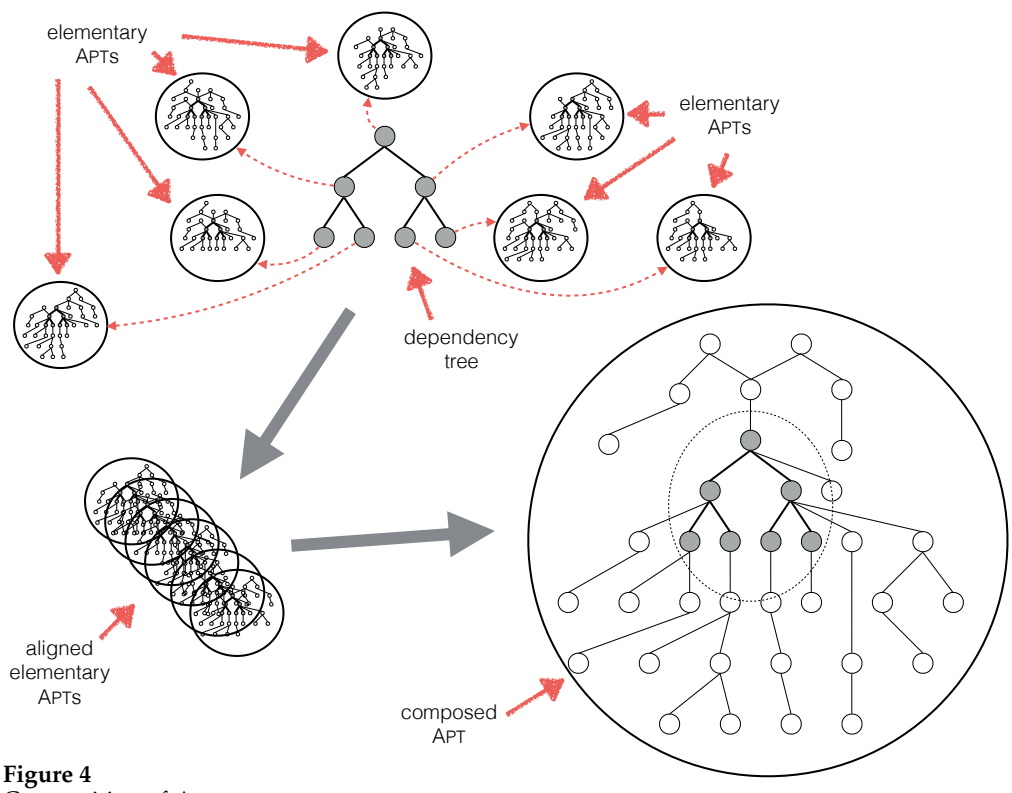

Figure 4
Composition of APTs.

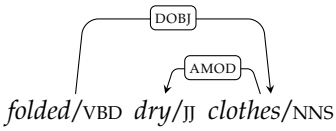

Figure 5
A dependency tree that generates the alignment shown in Figure 6.

the phrase *folded dry clothes*. To align the elementary APTs for the lexemes in this tree, we do the following:

- The anchor of the elementary APT for *dry*/JJ is moved to the node on which the *bought*/VBD and *folded*/VBD lie. This is the APT shown at the top of Figure 6. This change of anchor location is determined by the path from the *dry*/JJ to *folded*/VBD in the tree in Figure 5 (i.e., $\overline{\text{AMOD}} \cdot \overline{\text{DOBJ}}$).
- The anchor of the elementary APT for *clothes*/NNS is moved to the node on which *folded*/VBD, *hung*/VBD, and *bought*/VBD lie. This is the APT shown at the bottom of Figure 3. This change of anchor location is determined by the path from the *clothes*/NNS to *folded*/VBD in the tree in Figure 5 (i.e., $\overline{\text{DOBJ}}$).
- The anchor of the elementary APT for *folded*/VBD has been left unchanged because there is an empty path from *folded*/VBD to *folded*/VBD in the tree in Figure 5.

Figure 6 shows the three elementary APTs for the lexemes *dry*/JJ, *clothes*/NNS, and *folded*/VPD, which have been aligned as determined by the dependency tree shown in Figure 5. Each column of lexemes appears at nodes that have been aligned with one another. For example, in the third column from the left, we see that the following three nodes have been aligned: (i) the node in the elementary APT for *dry*/JJ at which *bought*/VBD and *folded*/VBD appear; (ii) the node in the elementary APT for *clothes*/NNS at which *folded*/VBD, *hung*/VBD, and *bought*/VBD appear; and (iii) the anchor node of the elementary APT for *folded*/VBD, that is, the node at which *folded*/VBD appears. In the second phase of composition, these three nodes are merged together to produce a single node in the composed APT.

Before we discuss how the nodes in aligned APTs are merged, we formalize the notion of APT alignment. We do this by first defining so-called offset APTs, which formalizes the idea of adjusting the location of an anchor. We then define how to align all of the APTs for the lexemes in a phrase based on a dependency tree.

4.2 Offset APTs

Given some offset, δ, a string in $\overline{R}^*R^*$, the APT $\mathbf{A}$ when offset by δ is denoted $\mathbf{A}^\delta$. Offsetting an APT by δ involves moving the anchor to the position reached by following the path δ from the original anchor position. In order to define $\mathbf{A}^\delta$, we must define $\mathbf{A}^\delta(\tau, w')$ for each $\tau \in \overline{R}^*R^*$ and $w' \in V$—or, in terms of our alternative tree-based representation, we need to specify the τ' such that $\mathbf{A}^\delta(\tau)$ and $\mathbf{A}(\tau')$ yield the same node (weighted lexeme multiset).

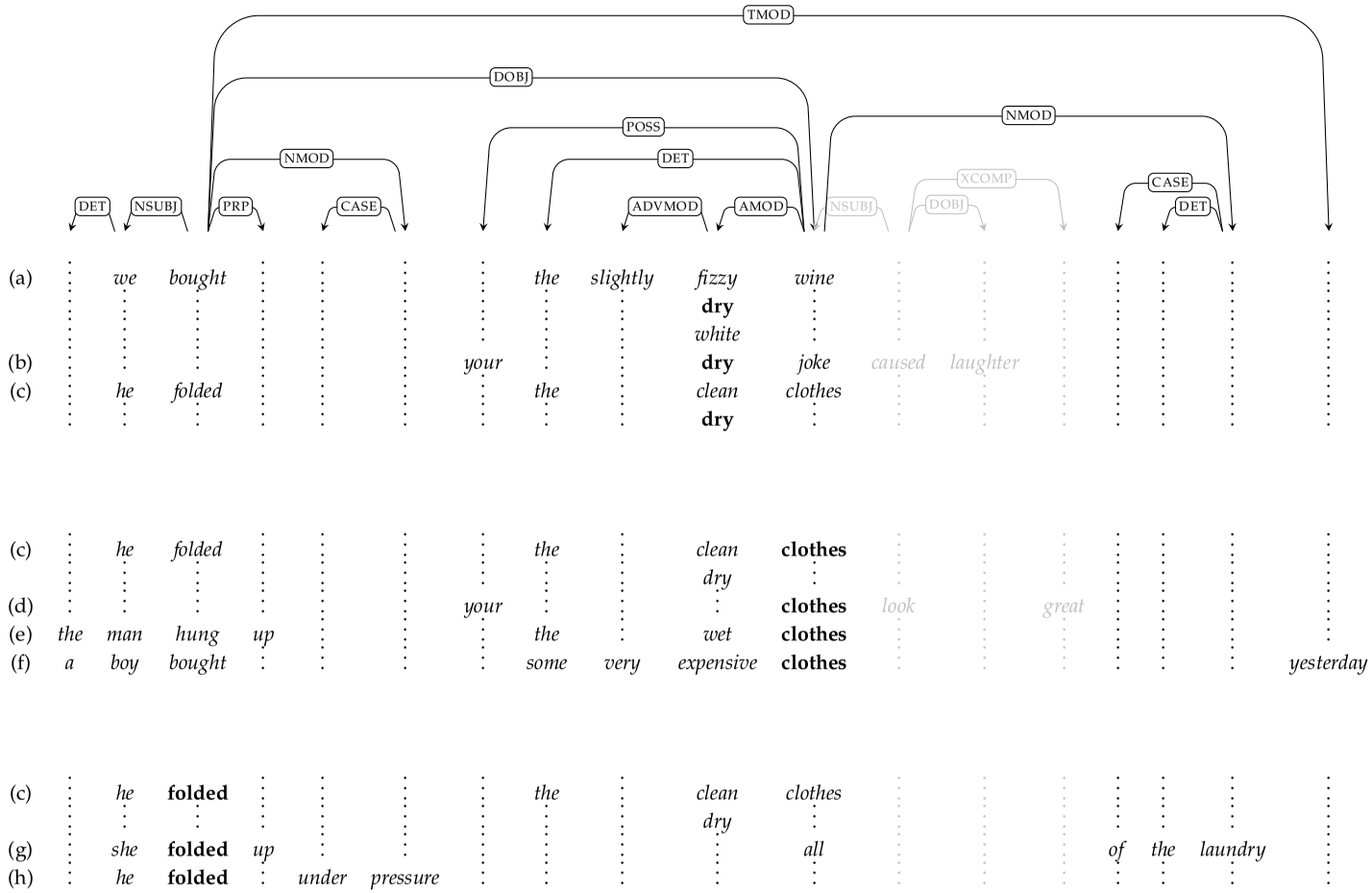

Figure 6
Vertically aligned APTs for *dry*/JJ, *clothes*/NNS, and *folded*/VBD as determined by the tree in Figure 5. The letters in brackets on the left refer to the dependency trees shown in Figure 1 from which these APTs are constructed. For space reasons, part-of-speech tags have been omitted.

As shown in Equation (7), path offset can be specified by making use of the co-occurrence type reduction operator that was introduced in Section 2.2. Given a string δ in $\overline{R}^*R^*$ and an APT $\mathbf{A}$, the offset APT $\mathbf{A}^\delta$ is defined as follows. For each $\tau \in \overline{R}^*R^*$ and $w \in V$:

$$\mathbf{A}^\delta(\tau, w) = \mathbf{A}(\downarrow(\delta\tau), w) \tag{7}$$

or equivalently, for each $\tau \in \overline{R}^*R^*$:

$$\mathbf{A}^\delta(\tau) = \mathbf{A}(\downarrow(\delta\tau)) \tag{8}$$

As required, Equation (7) defines $\mathbf{A}^\delta$ by specifying the weighted lexeme multiset we obtain when $\mathbf{A}^\delta$ is applied to co-occurrence type τ as being the lexeme multiset that $\mathbf{A}$ produces when applied to the co-occurrence type $\downarrow(\delta\tau)$.

As an illustrative example, consider the APT shown at the top of Figure 2. Let us call this APT $\mathbf{A}$. Note that $\mathbf{A}$ is anchored at the node where the lexeme *dry*/JJ appears. Consider the APT produced when we apply the offset $\overline{\text{AMOD}} \cdot \overline{\text{DOBJ}}$. This is shown at the top of Figure 3. Let us refer to this APT as $\mathbf{A}'$. The anchor of $\mathbf{A}'$ is the node at which the lexemes *bought*/VDB and *folded*/VBD appear. Now we show how the two nodes $\mathbf{A}'(\text{NSUBJ})$ and $\mathbf{A}'(\text{DOBJ}\cdot\text{AMOD}\cdot\text{ADVMOD})$ are defined in terms of $\mathbf{A}$ on the basis of Equation (8). In both cases the offset $\delta = \overline{\text{AMOD}} \cdot \overline{\text{DOBJ}}$.

- For the case where $\tau = \text{NSUBJ}$ we have

$$\begin{aligned}\mathbf{A}'(\text{NSUBJ}) &= \mathbf{A}(\downarrow(\overline{\text{AMOD}} \cdot \overline{\text{DOBJ}}\cdot\text{NSUBJ})) \\ &= \mathbf{A}(\overline{\text{AMOD}} \cdot \overline{\text{DOBJ}}\cdot\text{NSUBJ})\end{aligned}$$

  With respect to the anchor of $\mathbf{A}$, this correctly addresses the node at which the lexemes *we*/PRP and *he*/PRP appear.

- Where $\tau = \text{DOBJ}\cdot\text{AMOD}\cdot\text{ADVMOD}$ we have

$$\begin{aligned}\mathbf{A}'(\text{DOBJ}\cdot\text{AMOD}\cdot\text{ADVMOD}) &= \mathbf{A}(\downarrow(\overline{\text{AMOD}} \cdot \overline{\text{DOBJ}} \cdot \text{DOBJ} \cdot \text{AMOD} \cdot \text{ADVMOD})) \\ &= \mathbf{A}(\downarrow(\overline{\text{AMOD}} \cdot\text{AMOD}\cdot\text{ADVMOD})) \\ &= \mathbf{A}(\downarrow(\text{ADVMOD})) \\ &= \mathbf{A}(\text{ADVMOD})\end{aligned}$$

  With respect to the anchor of $\mathbf{A}$, this correctly addresses the node at which the lexeme *slightly*/RB appears.

In practice, the offset APT $\mathbf{A}^\delta$ can be obtained by prepending the inverse of the path offset, δ^{-1}, to all of the co-occurrence types in $\mathbf{A}$ and then repeatedly applying the reduction operator until no further reductions are possible. In other words, if τ addresses a node in $\mathbf{A}$, then τ' addresses a node in $\mathbf{A}^\delta$ iff $\tau' = \downarrow(\delta^{-1}\tau)$ and $\tau' \in \overline{R}^*R^*$.

4.3 Syntax-Driven APT Alignment

We now make use of offset APTs, as defined in Equation (7), as a way to align all of the APTs associated with a dependency tree. Consider the following scenario:

- $w_1 \ldots w_n$ is a the phrase (or sentence) where each $w_i \in V$ for $1 \leq i \leq n$
- $t \in T_{V,R}$ is a dependency analysis of the string $w_1 \ldots w_n$
- w_h is the lexeme at the root of t. In other words, h is the position (index) in the phrase at which the head appears
- $\|w_i\|$ is the elementary APT for w_i for each i, $1 \leq i \leq n$
- δ_i, the offset of w_i in t with respect to the root, is the path in t from w_i to w_h. In other words, $\langle w_i, \delta_i, w_h \rangle$ is a co-occurrence in t for each i, $1 \leq i \leq n$ (Note that $\delta_h = \epsilon$)

We define the distributional semantics for the tree t, denoted $\|t\|$, as follows:

$$\|t\| = \bigsqcup \left\{ \|w_1\|^{\delta_1}, \ldots, \|w_n\|^{\delta_n} \right\} \tag{9}$$

The definition of $\bigsqcup$ is considered in Section 4.4. In general, $\bigsqcup$ operates on a set of n aligned APTs, merging them into a single APT. The multiset at each node in the resulting APT is formed by merging n multisets, one from each of the elements of $\left\{ \|w_1\|^{\delta_1}, \ldots, \|w_n\|^{\delta_n} \right\}$. It is this multiset merging operation that we focus on in Section 4.4.

Although $\|t\|$ can be taken to be the distributional semantics of the tree as a whole, the same APT, when associated with different anchors (i.e., when offset in some appropriate way) provides a representation of each of the contextualized lexemes that appear in the tree.

For each i, for $1 \leq i \leq n$, the APT for w_i when contextualized by its role in the dependency tree t, denoted $\|w_i; t\|$, is the APT that satisfies the equality:

$$\|w_i; t\|^{\delta_i} = \|t\| \tag{10}$$

Alternatively, this can also be expressed with the equality:

$$\|w_i; t\| = \|t\|^{\delta_i{}^{-1}} \tag{11}$$

Note that $\|w_h; t\|$ and $\|t\|$ are identical. In other words, we take the representation of the distributional semantics of a dependency tree to be the APT for the lexeme at the root of that tree that has been contextualized by the other lexemes appearing below it in the tree.

Equation (9) defined APT composition as a "one-step" process in the sense that all of the n elementary APTs that are associated with nodes in the dependency tree are composed at once to produce the resulting (composed) APT. There are, however, alternative strategies that could be formulated. One possibility is fully incremental left-to-right composition, where, working left-to-right through the string of lexemes, the elementary APTs for the first two lexemes are composed, with the resulting APT then being composed with the elementary APT for the third lexeme, and so on. It is always

possible to compose APTs in this fully incremental way, whatever the structure in the dependency tree. The tree structure, however, is critical in determining how the adjacent APTs need to be aligned.

4.4 Merging Aligned APTs

We now turn to the question of how to implement the function $\bigsqcup$ that appears in Equation (9). $\bigsqcup$ takes a set of n aligned APTs, $\{\mathbf{A}_1, \ldots \mathbf{A}_n\}$, one for each node in the dependency tree t. It merges the APTs together node by node to produce a single APT, $\bigsqcup\{\mathbf{A}_1, \ldots \mathbf{A}_n\}$, that represents the semantics of the dependency tree. Our discussion, therefore, addresses the question of how to merge the multisets that appear at nodes that are aligned with each other and form the nodes of the APT being produced.

The elementary APT for a lexeme expresses those co-occurrences that are **distributionally compatible** with the lexeme given the corpus. When lexemes in some phrase are composed, our objective is to capture the extent to which the co-occurrences arising in the elementary APTs are mutually compatible with the phrase as a whole. Once the elementary APTs that are being composed have been aligned, we are in a position to determine the extent to which co-occurrences are mutually compatible: Co-occurrences that need to be compatible with one another are brought together through the alignment. We consider two alternative ways in which this can be achieved.

We begin with $\bigsqcup_{\text{INT}}$, which provides a tight implementation of the mutual compatibility of co-occurrences. In particular, a co-occurrence is only deemed to be compatible with the composed lexemes to the extent that is distributionally compatible with the lexeme that it is least compatible with. This corresponds to the multiset version of intersection. In particular, for all $\tau \in \overline{R}^* R^*$ and $w' \in V$:

$$\bigsqcup_{\text{INT}} \{\mathbf{A}_1, \ldots, \mathbf{A}_n\}(\tau, w') = \min_{1 \leq i \leq n} \mathbf{A}_i(\tau, w') \tag{12}$$

It is clear that the effectiveness of $\bigsqcup_{\text{INT}}$ increases as the size of C grows, and that it would particularly benefit from distributional smoothing (Dagan, Pereira, and Lee 1994), which can be used to improve plausible co-occurrence coverage by inferring co-occurrences in the APT for a lexeme w based on the co-occurrences in the APTs of distributionally similar lexemes.

An alternative to $\bigsqcup_{\text{INT}}$ is $\bigsqcup_{\text{UNI}}$, where we determine distributional compatibility of a co-occurrence by aggregating across the distributional compatibility of the co-occurrence for each of the lexemes being composed. In particular, for all $\tau \in (\overline{R} \cup R)^*$ and $w' \in V$:

$$\bigsqcup_{\text{UNI}} \{\mathbf{A}_1, \ldots, \mathbf{A}_n\}(\tau, w') = \sum_{1 \leq i \leq n} \mathbf{A}_i(\tau, w') \tag{13}$$

Although this clearly achieves co-occurrence embellishment, whether co-occurrence filtering is achieved depends on the weighting scheme being used. For example, if negative weights are allowed, then co-occurrence filtering can be achieved.

There is one very important feature of APT composition that is a distinctive aspect of our proposal, and therefore worth dwelling on. In Section 4.1, when discussing Figure 4, we made reference to the notions of internal and external context. The internal context of a composed APT is that part of the APT that corresponds to the nodes in the dependency tree that generated the composed APT. One might have expected that

the only lexeme appearing at an internal node is the lexeme that appears at the corresponding node in the dependency tree. However, this is absolutely not the objective: At each node in the internal context, we expect to find a set of alternative lexemes that are, to varying degrees, distributionally compatible with that position in the APT. We expect that a lexeme that is distributionally compatible with a substantial number of the lexemes being composed will result in a distributional feature with non-zero weight in the vectorized APT. There is, therefore, no distinction being made between internal and external nodes. This enriches the distributional representation of the contextualized lexemes, and overcomes the potential problem arising from the fact that as larger and larger units are composed, there is less and less external context around to characterize distributional meaning.

5. Experiments

In this section we consider some empirical evidence in support of APTs. First, we consider some of the different ways in which APTs can be instantiated. Second, we present a number of case studies showing the disambiguating effect of APT composition in adjective–noun composition. Finally, we evaluate the model using the phrase-based compositionality benchmarks of Mitchell and Lapata (2008, 2010).

5.1 Instantiating APTs

We have constructed APT lexicons from three different corpora.

- `clean_wiki` is a corpus used for the case studies in Section 5.2. This corpus is a cleaned 2013 Wikipedia dump (Wilson 2015) that we have tokenized, part-of-speech-tagged, lemmatized, and dependency-parsed using the Malt Parser (Nivre 2004). This corpus contains approximately 0.6 billion tokens.
- `BNC` is the British National Corpus. It has been tokenized, POS-tagged, lemmatized, and dependency-parsed as described in Grefenstette et al. (2013) and contains approximately 0.1 billion tokens.
- `concat` is a concatenation of the ukWaC corpus (Ferraresi et al. 2008), a mid-2009 dump of the English Wikipedia and the British National Corpus. This corpus has been tokenized, POS-tagged, lemmatized, and dependency-parsed as described in Grefenstette et al. (2013) and contains about 2.8 billion tokens.

Having constructed lexicons, there are a number of hyperparameters to be explored during composition. First there is the composition operation itself. We have explored variants that take a union of the features such as `add` and `max` and variants that take an intersection of the features such as `mult`, `min`, and `intersective_add`, where $\texttt{intersective_add}(a, b) = a + b$ iff $a > 0$ and $b > 0$; 0 otherwise.

Second, the APT theory is agnostic to the type or derivation of the weights that are being composed. The weights in the elementary APTs can be counts, probabilities, or some variant of PPMI or other association function. Although it is generally accepted that the use of some association function such as PPMI is normally beneficial in the determination of lexical similarity, there is a choice over whether these weights should be seen as part of the representation of the lexeme, or as part of the similarity calculation.

In the instantiation that we refer to as as `compose_first`, APT weights are probabilities. These are composed and transformed to PPMI scores before computing cosine similarities. In the instantiation that we refer to as `compose_second`, APT weights are PPMI scores.

There are a number of modifications that can be made to the standard PPMI calculation. First, it is common (Levy, Goldberg, and Dagan 2015) to delete rare words when building co-occurrence vectors. Low-frequency features contribute little to similarity calculations because they co-occur with very few of the targets. Their inclusion will tend to reduce similarity scores across the board, but have little effect on ranking. Filtering, on the other hand, improves efficiency. In other experiments, we have found that a feature frequency threshold of 1,000 works well. On a corpus the size of Wikipedia (1.5 billion tokens), this leads to a feature space for nouns of approximately 80,000 dimensions (when including only first-order paths) and approximately 230,000 dimensions (when including paths up to order 2).

Levy, Goldberg, and Dagan (2015) also showed that the use of context distribution smoothing (`cds`), $\alpha = 0.75$, can lead to performance comparable with state-of-the-art word embeddings on word similarity tasks.

$$\text{PMI}_\alpha\left(w', w; \tau\right) = \log \frac{\#\langle w,\ \tau,\ w'\rangle\ \#\langle *,\ \tau,\ *\rangle^\alpha}{\#\langle w,\ \tau,\ *\rangle\ \#\langle *,\ \tau,\ w'\rangle^\alpha}$$

Levy, Goldberg, and Dagan (2015) further showed that using shifted PMI, which is analogous to the use of negative sampling in word embeddings, can be advantageous. When shifting PMI, all values are shifted down by $\log k$ before the threshold is applied.

$$\text{SPPMI}\left(w', w; \tau\right) = \max\left(\text{PMI}\left(w', w; \tau\right) - \log k, 0\right)$$

Finally, there are many possible options for the path weighting function $\phi(\tau, w)$. These include the path probability $p(\tau \mid w)$ as discussed in Section 3, constant path weighting, and inverse path length or harmonic function (which is equivalent to the dynamic context window used in many neural implementations such as GloVe [Pennington, Socher, and Manning 2014]).

5.2 Disambiguation

Here we consider the differences between using aligned and unaligned APT representations as well as the differences between using $\bigsqcup_{\text{UNI}}$ and $\bigsqcup_{\text{INT}}$ when carrying out adjective–noun (AN) composition. From the `clean_wiki` corpus described in Section 5.1, a small number of high-frequency nouns were chosen that are ambiguous or broad in meaning together with potentially disambiguating adjectives. We use the `compose_first` option where composition is carried out on APTs containing probabilities.

$$\text{W}(w,\ \langle\ \tau, w'\ \rangle) = \frac{\#\langle w,\ \tau,\ w'\rangle}{\#\langle w,\ *,\ *\rangle}$$

The closest distributional neighbors of the individual lexemes before and after composition with the disambiguating adjective are then examined. In order to calculate similarities, contexts are weighted using the variant of PPMI advocated by Levy,

Goldberg, and Dagan (2015), where cds is applied with $\alpha = 0.75$. However, no shift is applied to the PPMI values because we have found shifting to have little or negative effect when working with relatively small corpora. Similarity is then computed using the standard cosine measure. For illustrative purposes the top ten neighbors of each word or phrase are shown, concentrating on ranks rather than absolute similarity scores.

Table 2 illustrates what happens when $\bigsqcup_{\text{UNI}}$ is used to merge aligned and unaligned APT representations when the noun *shoot* is placed in the contexts of *green* and *six-week*. Boldface is used in the entries of compounds where a neighbor appears to be highly suggestive of the intended sense and where it has a rank higher or equal to its rank in the entry for the uncontextualized noun. In this example, it is clear that merging the unaligned APT representations provides very little disambiguation of the target noun. This is because typed co-occurrences for an adjective mostly belong in a different space to typed co-occurrences for a noun. Addition of these spaces leads to significantly lower absolute similarity scores, but little change in the ranking of neighbors. Although we only show one example here, this observation appears to hold true whenever words with different part of speech tags are composed. Intersection of these spaces via $\bigsqcup_{\text{INT}}$ generally leads to substantially degraded neighbors, often little better than random, as illustrated by Table 3.

On the other hand, when APTs are correctly aligned and merged using $\bigsqcup_{\text{UNI}}$, we see the disambiguating effect of the adjective. A *green shoot* is more similar to *leaf*, *flower*, *fruit*, and *tree*. A *six-week shoot* is more similar to *tour*, *session*, *show*, and *concert*. This disambiguating effect is even more apparent when $\bigsqcup_{\text{INT}}$ is used to merge the APT representations (see Table 3).

Table 4 further illustrates the difference between using $\bigsqcup_{\text{UNI}}$ and $\bigsqcup_{\text{INT}}$ when composing aligned APT representations. Again, boldface is used in the entries of compounds where a neighbor appears to be highly suggestive of the intended sense and where it has a rank higher or equal to its rank in the entry for the uncontextualized noun. In these examples, we can see that both $\bigsqcup_{\text{UNI}}$ and $\bigsqcup_{\text{INT}}$ appear to be effective in carrying out some disambiguation. Looking at the example of *musical group*, both $\bigsqcup_{\text{UNI}}$ and $\bigsqcup_{\text{INT}}$ increase the relative similarity of *band* and *music* to *group* when it is contextualized by *musical*. However, $\bigsqcup_{\text{INT}}$ also leads to a number of other words being selected as neighbors that

Table 2
Neighbors of uncontextualized shoot/N compared with shoot/N in the contexts of green/J and six-week/J, using $\bigsqcup_{\text{UNI}}$ with aligned and unaligned representations.

	Aligned $\bigsqcup_{\text{UNI}}$		Unaligned $\bigsqcup_{\text{UNI}}$	
shoot	green shoot	six-week shoot	green shoot	six-week shoot
shot	shoot	shoot	shoot	shoot
leaf	**leaf**	**tour**	shot	shot
shooting	**flower**	shot	leaf	shooting
fight	**fruit**	**break**	shooting	leaf
scene	orange	**session**	fight	scene
video	**tree**	**show**	scene	**video**
tour	color	shooting	video	fight
footage	shot	**concert**	tour	footage
interview	color	**interview**	flower	**photo**
flower	cover	leaf	footage	interview

Table 3
Neighbors of uncontextualized shoot/N compared with shoot/N in the contexts of green/J and six-week/J, using $\bigsqcup_{\text{INT}}$ with aligned and unaligned representations.

	Aligned $\bigsqcup_{\text{INT}}$		Unaligned $\bigsqcup_{\text{INT}}$	
shoot	green shoot	six-week shoot	green shoot	six-week shoot
shot	shoot	shoot	shoot	e/f
leaf	**leaf**	**photoshoot**	pyrite	uemtsu
shooting	**fruit**	**taping**	plosive	confederations
fight	**stalk**	**tour**	handlebars	shortlist
scene	**flower**	**airing**	annual	all-ireland
video	**twig**	**rehearsal**	roundel	dern
tour	**sprout**	**broadcast**	affricate	gerwen
flower	**bud**	**session**	phosphor	tactics
footage	**shrub**	q&a	connections	backstroke
interview	**inflorescence**	post-production	reduplication	gabler

are closely related to the musical sense of group (e.g., *troupe*, *ensemble*, and *trio*). This is not the case when $\bigsqcup_{\text{UNI}}$ is used—the other neighbors still appear related to the general meaning of *group*. This trend is also seen in some of the other examples such as *ethnic group*, *human body*, and *magnetic field*. Further, even when $\bigsqcup_{\text{UNI}}$ leads to the successful selection of a large number of sense specific neighbors (e.g., see *literary work)*, the neighbors selected appear to be higher frequency, more general words than when $\bigsqcup_{\text{INT}}$ is used.

The reason for this is likely to be the effect that each of these composition operations has on the number of non-zero dimensions in the composed representations. Ignoring the relatively small effect the feature association function may have on this, it is obvious that $\bigsqcup_{\text{UNI}}$ should increase the number of non-zero dimensions, whereas $\bigsqcup_{\text{INT}}$ should decrease the number of non-zero dimensions. In general, the number of non-zero dimensions is highly correlated with frequency, which makes composed representations based on $\bigsqcup_{\text{UNI}}$ behave like high-frequency words and composed representations based on $\bigsqcup_{\text{INT}}$ behave like low-frequency words. Further, when using similarity measures based on PPMI, as demonstrated by Weeds (2003), it is not unusual to find that the neighbors of high-frequency entities (with a large number of non-zero dimensions) are other high-frequency entities (also with a large number of non-zero dimensions). Nor is it unusual to find that the neighbors of low-frequency entities (with a small number of non-zero dimensions) are other low-frequency entities (with a small number of non-zero dimensions). Weeds, Weir, and McCarthy (2004) showed that frequency is also a surprisingly good indicator of the generality of the word. Hence $\bigsqcup_{\text{UNI}}$ leads to more general neighbors and $\bigsqcup_{\text{INT}}$ leads to more specific neighbors.

Finally, note that whereas $\bigsqcup_{\text{INT}}$ has produced high quality neighbors in these examples where only two words are composed, using $\bigsqcup_{\text{INT}}$ in the context of the composition of an entire sentence would tend to lead to very sparse representations. The majority of the internal nodes of the APT composed using an intersective operation such as $\bigsqcup_{\text{INT}}$ must necessarily only include the lexemes actually used in the sentence. $\bigsqcup_{\text{UNI}}$, on the other hand, will have added to these internal representations, suggesting similar words that might have been used in those contexts and giving rise to a rich representation that might be used to calculate sentence similarity. Further, the use of PPMI, or some other similar form of feature weighting and selection, will mean

Table 4
Distributional neighbors using $\bigsqcup_{\text{UNI}}$ vs $\bigsqcup_{\text{INT}}$ (e-magnetism = electro-magnetism).

	Aligned $\bigsqcup_{\text{UNI}}$		Unaligned Aligned $\bigsqcup_{\text{INT}}$	
group	musical group	ethnic group	musical group	ethnic group
group	group	group	group	group
organization	company	organization	**band**	**community**
organisation	**band**	organisation	**troupe**	organization
company	**music**	**community**	**ensemble**	grouping
community	movement	company	**artist**	sub-group
corporation	community	movement	**trio**	**faction**
unit	society	society	genre	**ethnicity**
movement	corporation	**minority**	**music**	**minority**
association	category	unit	**duo**	organisation
society	association	entity	**supergroup**	**tribe**
body	human body	legislative body	human body	legislative body
body	body	body	body	body
board	organization	**council**	**organism**	**council**
organization	structure	**committee**	organization	**committee**
entity	entity	board	entity	board
skin	organisation	**authority**	**embryo**	**legislature**
head	skin	**assembly**	**brain**	**secretariat**
organisation	**brain**	organisation	community	**authority**
structure	**eye**	**agency**	**organelle**	**assembly**
council	object	**commission**	institution	**power**
eye	**organ**	entity	**cranium**	**office**
work	social work	literary work	social work	literary work
study	work	work	work	work
project	activity	**book**	research	**writings**
book	study	study	study	**treatise**
activity	project	**novel**	writings	**essay**
effort	program	project	endeavour	**poem**
publication	practice	**publication**	project	book
job	**development**	**text**	discourse	**novel**
program	aspect	**literature**	topic	**monograph**
writing	book	**story**	**development**	**poetry**
piece	effort	**writing**	**teaching**	**writing**
field	athletic field	magnetic field	athletic field	magnetic field
facility	field	field	field	field
stadium	facility	**component**	**gymnasium**	**wavefunction**
area	**stadium**	stadium	**fieldhouse**	**spacetime**
complex	**gymnasium**	facility	**stadium**	**flux**
ground	**basketball**	track	**gym**	subfield
pool	**sport**	ground	**arena**	**perturbation**
base	**center**	system	**rink**	**vector**
space	**softball**	complex	**softball**	**e-magnetism**
centre	**gym**	**parameter**	cafeteria	formula_8
park	**arena**	pool	**ballpark**	scalar

that those internal (and external) contexts that are not supported by a majority of the lexemes in the sentence will tend to be considered insignificant and therefore will be ignored in similarity calculations. By using shifted PPMI, it should be possible to further reduce the number of non-zero dimensions in a representation constructed using $\bigsqcup_{\text{UNI}}$, and this should also allow us to control the specificity/generality of the neighbors observed.

5.3 Phrase-Based Composition Tasks

Here we look at the performance of one instantiation of the APT framework on two benchmark tasks for phrase-based composition.

5.3.1 Experiment 1: The M&L2010 Data Set. The first experiment uses the M&L2010 data set, introduced by Mitchell and Lapata (2010), which contains human similarity judgments for adjective–noun (AN), noun–noun (NN), and verb–object (VO) combinations on a seven-point rating scale. It contains 108 combinations in each category such as $\langle$*social activity*, *economic condition*$\rangle$, $\langle$*tv set*, *bedroom window*$\rangle$, and $\langle$*fight war*, *win battle*$\rangle$. This data set has been used in a number of evaluations of compositional methods including Mitchell and Lapata (2010), Blacoe and Lapata (2012), Turney (2012), Hermann and Blunsom (2013), and Kiela and Clark (2014). For example, Blacoe and Lapata (2012) show that multiplication in a simple distributional space (referred to here as an *untyped VSM*) outperforms the distributional memory (DM) method of Baroni and Lenci (2010) and the neural language model (NLM) method of Collobert and Weston (2008).

Although often not explicit, the experimental procedure in most of this work would appear to be the calculation of Spearman's rank correlation coefficient ρ between model scores and individual, non-aggregated, human ratings. For example, if there are 108 phrase pairs being judged by 6 humans, this would lead to a data set containing 648 data points. The procedure is discussed at length in Turney (2012), who argues that this method tends to underestimate model performance. Accordingly, Turney explicitly uses a different procedure where a separate Spearman's ρ is calculated between the model scores and the scores of each participant. These coefficients are then averaged to give the performance indicator for each model. Here, we report results using the original `M&L` method (see Table 5). We found that using the Turney method, scores were typically higher by 0.01 to 0.04. If model scores are evaluated against aggregated human scores, then the values of Spearman's ρ tend to be still higher, typically 0.1 to 0.12 higher than the values reported here.

For this experiment, we have constructed an order 2 APT lexicon for the `BNC` corpus. This is the same corpus used by Mitchell and Lapata (2010) and for the best performing algorithms in Blacoe and Lapata (2012). We note that the larger `concat` corpus was used by Blacoe and Lapata (2012) in the evaluation of the DM algorithm (Baroni and Lenci 2010). We use the `compose_second` option, where the elementary APT weights are PPMI. With regard to the different parameter settings in the PPMI calculation (Levy, Goldberg, and Dagan 2015), we tuned on a number of popular word similarity tasks: MEN (Bruni, Tran, and Baroni 2014); WordSim-353 (Finkelstein et al. 2001); and SimLex-999 (Hill, Reichart, and Korhonen 2015). In these tuning experiments, we found that context distribution smoothing gave mixed results. However, shifting PPMI ($k = 10$) gave optimal results across all of the word similarity tasks. Therefore we report results here for vanilla PPMI (shift $k = 1$) and shifted PPMI (shift $k = 10$). For composition, we report results for both $\bigsqcup_{\text{UNI}}$ and $\bigsqcup_{\text{INT}}$. Results are shown in Table 5.

Table 5
Results on the M&L2010 data set using the `M&L` method of evaluation. Values shown are Spearman's ρ.

	AN	NN	VO	Average
$\bigsqcup_{\text{INT}}, k = 1$	−0.09	0.43	0.35	0.23
$\bigsqcup_{\text{INT}}, k = 10$	NaN	0.23	0.26	0.16
$\bigsqcup_{\text{UNI}}, k = 1$	0.47	0.37	0.40	0.41
$\bigsqcup_{\text{UNI}}, k = 10$	0.45	0.42	**0.42**	0.43
untyped VSM, `multiply` (Mitchell and Lapata 2010)	0.46	0.49	0.37	**0.44**
untyped VSM, `multiply` (Blacoe and Lapata 2012)	**0.48**	**0.50**	0.35	**0.44**
distributional memory (DM), `add` (Blacoe and Lapata 2012)	0.37	0.30	0.29	0.32
neural language model (NLM), `add` (Blacoe and Lapata 2012)	0.28	0.26	0.24	0.26
humans (Mitchell and Lapata 2010)	0.52	0.49	0.55	0.52

For this task and with this corpus, $\bigsqcup_{\text{UNI}}$ consistently outperforms $\bigsqcup_{\text{INT}}$. Shifting PPMI by $\log 10$ consistently improves results for $\bigsqcup_{\text{UNI}}$, but has a large negative effect on the results for $\bigsqcup_{\text{INT}}$. We believe that this is due to the relatively small size of the corpus. Shifting PPMI reduces the number of non-zero dimensions in each vector, which increases the likelihood of a zero intersection. In the case of AN composition, all of the intersections were zero for this setting, making it impossible to compute a correlation.

Comparing these results with the state of the art, we can see that $\bigsqcup_{\text{UNI}}$ clearly outperforms DM and NLM as tested by Blacoe and Lapata (2012). This method of composition also achieves close to the best results in Mitchell and Lapata (2010) and Blacoe and Lapata (2012). It is interesting to note that our model does substantially better than the state of the art on verb–object composition, but is considerably worse at noun–noun composition. Exploring why this is so is a matter for future research. We have undertaken experiments with a larger corpus and a larger range of hyper-parameter settings, which indicate that the performance of the APT models can be increased significantly. However, these results are not presented here, because an equitable comparison with existing models would require a similar exploration of the hyper-parameter space across all models being compared.

5.3.2 Experiment 2: The M&L2008 Data Set. The second experiment uses the M&L2008 data set, introduced by Mitchell and Lapata (2008), which contains pairs of intransitive sensitives together with human judgments of similarity. The data set contains 120 unique subject, verb, landmark triples with a varying number of human judgments per item. On average each triple is rated by 30 participants. The task is to rate the similarity of the verb and the landmark given the potentially disambiguating context of the subject. For example, in the context of the subject *fire* one might expect *glowed* to be close to *burned* but not close to *beamed*. Conversely, in the context of the subject *face* one might expect *glowed* to be close to *beamed* and not close to *burned*.

This data set was used in the evaluations carried out by Grefenstette et al. (2013) and Dinu, Pham, and Baroni (2013). These evaluations clearly follow the experimental procedure of Mitchell and Lapata and do not evaluate against mean scores. Instead, separate points are created for each human annotator, as discussed in Section 5.3.1.

The multi-step regression algorithm of Grefenstette et al. (2013) achieved $\rho = 0.23$ on this data set. In the evaluation of Dinu, Pham, and Baroni (2013), the lexical function algorithm, which learns a matrix representation for each functor and defines composition as matrix-vector multiplication, was the best-performing compositional algorithm at this task. With optimal parameter settings, it achieved around $\rho = 0.26$. In this evaluation, the full additive model of Guevara (2010) achieved $\rho < 0.05$.

In order to make our results directly comparable with these previous evaluations, we used the same corpus to construct our APT lexicons, namely, the `concat` corpus described in Section 5.1. Otherwise, the APT lexicon was constructed as described in Section 5.3.1. As before, note that $k = 1$ in shifted PPMI is equivalent to not shifting PPMI. Results are shown in Table 6.

We see that $\bigsqcup_{\text{UNI}}$ is highly competitive with the optimized lexical function model that was the best performing model in the evaluation of Dinu, Pham, and Baroni (2013). In that evaluation, the lexical function model achieved between 0.23 and 0.26, depending on the parameters used in dimensionality reduction. Using vanilla PPMI, without any context distribution smoothing or shifting, $\bigsqcup_{\text{UNI}}$ achieves $\rho = 0.20$, which is less than $\bigsqcup_{\text{INT}}$. However, when using shifted PPMI as weights, the best result is 0.26. The shifting of PPMI means that contexts need to be more surprising in order to be considered as features. This makes sense when using an additive model such as $\bigsqcup_{\text{UNI}}$.

We also see that at this task and using this corpus, $\bigsqcup_{\text{INT}}$ performs relatively well. Using vanilla PPMI, without any context distribution smoothing or shifting, it achieves $\rho = 0.23$, which equals the performance of the multi-step regression algorithm of Grefenstette et al. (2013). Here, however, shifting PPMI has a negative impact on performance. This is largely because of the intersective nature of the composition

Table 6
Results on the M&L2008 data set. Values shown are Spearman's ρ.

$\bigsqcup_{\text{INT}}, k = 1$	0.23
$\bigsqcup_{\text{INT}}, k = 10$	0.13
$\bigsqcup_{\text{UNI}}, k = 1$	0.20
$\bigsqcup_{\text{UNI}}, k = 10$	**0.26**
multi-step regression Grefenstette et al. (2013)	0.23
lexical function Dinu, Pham, and Baroni (2013)	0.23–**0.26**
untyped VSM, `mult` Dinu, Pham, and Baroni (2013)	0.20–0.22
full additive Dinu, Pham, and Baroni (2013)	0–0.05
humans Mitchell and Lapata (2008)	0.40

operation—if shifting PPMI removes a feature from one of the unigram representations, it cannot be recovered during composition.

6. Related Work

Our work brings together two strands usually treated as separate though related problems: representing phrasal meaning by creating distributional representations through composition; and representing word meaning in context by modifying the distributional representation of a word. In common with some other work on lexical distributional similarity, we use a typed co-occurrence space. However, we propose the use of higher-order grammatical dependency relations to enable the representation of phrasal meaning and the representation of word meaning in context.

6.1 Representing Phrasal Meaning

The problem of representing phrasal meaning has traditionally been tackled by taking vector representations for words (Turney and Pantel 2010) and combining them using some function to produce a data structure that represents the phrase or sentence. Mitchell and Lapata (2008, 2010) found that simple additive and multiplicative functions applied to proximity-based vector representations were no less effective than more complex functions when performance was assessed against human similarity judgments of simple paired phrases.

The word embeddings learned by the continuous bag-of-words model (CBOW) and the continuous skip-gram model proposed by Mikolov et al. (2013a, 2013b) are currently among the most popular forms of distributional word representations. Although using a neural network architecture, the intuitions behind such distributed representations of words are the same as in traditional distributional representations. As argued by Pennington et al. (2014), both count-based and prediction-based models probe the underlying corpus co-occurrences statistics. For example, the CBOW architecture predicts the current word based on context (which is viewed as a bag-of-words) and the skip-gram architecture predicts surrounding words given the current word. Mikolov et al. (2013c) showed that it is possible to use these models to efficiently learn low-dimensional representations for words that appear to capture both syntactic and semantic regularities. Mikolov et al. (2013b) also demonstrated the possibility of composing skip-gram representations using addition. For example, they found that adding the vectors for *Russian* and *river* results in a very similar vector to the result of adding the vectors for *Volga* and *river*. This is similar to the multiplicative model of Mitchell and Lapata (2008) since the sum of two skip-gram word vectors is related to the product of two word context distributions.

Although our model shares with these the use of vector addition as a composition operation, the underlying framework is very different. Specifically, the actual vectors added depend not just on the form of the words but also their grammatical relationship within the phrase or sentence. This means that the representation for, say, *glass window* is not equal to the representation of *window glass*. The direction of the NN relationship between the words leads to a different alignment of the APTs and consequently a different representation for the phrases.

There are other approaches that incorporate theoretical ideas from formal semantics and machine learning, use syntactic information, and specialize the data structures to the task in hand. For adjective–noun phrase composition, Baroni and Zamparelli (2010)

and Guevara (2010) borrowed from formal semantics the notion that an adjective acts as a modifying function on the noun. They represented a noun as a vector, an adjective as a matrix, which could be induced from pairs of nouns and adjective noun phrases, and composed the two using matrix-by-vector multiplication to produce a vector for the noun phrase. Separately, Coecke, Sadrzadeh, and Clark (2011) proposed a broader compositional framework that incorporated from formal semantics the notion of function application derived from syntactic structure (Montague 1970; Lambek 1999). These two approaches were subsequently combined and extended to incorporate simple transitive and intransitive sentences, with functions represented by tensors, and arguments represented by vectors (Grefenstette et al. 2013).

The MV-RNN model of Socher et al. (2012) broadened the Baroni and Zamparelli (2010) approach; all words, regardless of part of speech, were modeled with both a vector and a matrix. This approach also shared features with Coecke, Sadrzadeh, and Clark (2011) in using syntax to guide the order of phrasal composition. This model, however, was made much more flexible by requiring and using task-specific labeled training data to create task-specific distributional data structures, and by allowing nonlinear relationships between component data structures and the composed result. The payoff for this increased flexibility has come with impressive performance in sentiment analysis (Socher et al. 2012, 2013).

However, although these approaches all pay attention to syntax, they all require large amounts of training data. For example, running regression models to accurately predict the matrix or tensor for each individual adjective or verb requires a large number of exemplar compositions containing that adjective or verb. Socher's MV-RNN model further requires task-specific labeled training data. Our approach, on the other hand, is purely count-based and directly aggregates information about each word from the corpus.

Other approaches have been proposed. Clarke (2007, 2012) suggested a context-theoretic semantic framework, incorporating a generative model that assigned probabilities to arbitrary word sequences. This approach shared with Coecke, Sadrzadeh, and Clark (2011) an ambition to provide a bridge between compositional distributional semantics and formal logic-based semantics. In a similar vein, Garrette, Erk, and Mooney (2011) combined word-level distributional vector representations with logic-based representation using a probabilistic reasoning framework. Lewis and Steedman (2013) also attempted to combine distributional and logical semantics by learning a lexicon for Combinatory Categorial Grammar (CCG; Steedman 2000), which first maps natural language to a deterministic logical form and then performs a distributional clustering over logical predicates based on arguments. The CCG formalism was also used by Hermann and Blunsom (2013) as a means for incorporating syntax-sensitivity into vector space representations of sentential semantics based on recursive auto-encoders (Socher et al. 2011a, 2011b). They achieved this by representing each combinatory step in a CCG parse tree with an auto-encoder function, where it is possible to parameterize both the weight matrix and bias on the combinatory rule and the CCG category.

Turney (2012) offered a model that incorporated assessments of word-level semantic relations in order to determine phrasal-level similarity. This work uses two different word-level distributional representations to encapsulate two types of similarity, and captures instances where the components of a composed noun phrase bore similarity to another word through a mix of those similarity types. Crucially, it views similarity of phrases as a function of the similarities of the components and does not attempt to derive modified vectors for phrases or words in context. Dinu and Thater (2012) also compared computing sentence similarity via additive compositional models with an

alignment-based approach, where sentence similarity is a function of the similarities of component words, and simple word overlap. Their results showed that a model based on a mixture of these approaches outperformed all of the individual approaches on a number of textual entailment data sets.

6.2 Typed Co-occurrence Models

In untyped co-occurrence models, such as those considered by Mitchell and Lapata (2008, 2010), co-occurrences are simple, untyped pairs of words that co-occur together (usually within some window of proximity but possibly within some grammatical relation). The lack of typing makes it possible to compose vectors through addition and multiplication. However, in the computation of lexical distributional similarity using grammatical dependency relations, it has been typical (Lin 1998; Lee 1999; Weeds and Weir 2005) to consider the *type* of a co-occurrence (for example, does *dog* occur with *eat* as its direct object or its subject?) as part of the feature space. The distinction between vector spaces based on untyped and typed co-occurrences was formalized by Padó and Lapata (2007) and Baroni and Lenci (2010). In particular, Baroni and Lenci showed that typed co-occurrences based on grammatical relations were better than untyped co-occurrences for distinguishing certain semantic relations. However, as shown by Weeds, Weir, and Reffin (2014), it does not make sense to compose typed features based on first-order dependency relations through multiplication and addition, because the vector spaces for different parts of speech are largely non-overlapping.

Padó and Lapata (2007) constructed features using higher-order grammatical dependency relations. They defined a path through a dependency tree in terms of the node words. This allowed words that are only indirectly related within a sentence to be considered as co-occurring. For example, in *a lorry carries apples*, there is a path of length 2 between the nouns *lorry* and *apples* via the node *carry*. However, they also used a word-based basis mapping, which essentially reduces all of the salient grammatical paths to untyped co-occurrences. Given the paths $\langle lorry, carry \rangle$ and $\langle lorry, carry, apples \rangle$ for *lorry*, these would be mapped to the basis elements *carry* and *apples*, respectively.

6.3 Representing Word Meaning in Context

A long-standing topic in distributional semantics has been the modification of a canonical representation of a lexeme's meaning to reflect the context in which it is found. Typically, a canonical vector for a lexeme is estimated from all corpus occurrences and the vector then modified to reflect the instance context (Lund and Burgess 1996; Erk and Padó 2008; Mitchell and Lapata 2008; Thater, Dinu, and Pinkal 2009; Thater, Fürstenau, and Pinkal 2010, 2011; Van de Cruys, Poibeau, and Korhonen 2011; Erk 2012).

As described in Mitchell and Lapata (2008, 2010), lexeme vectors have typically been modified using simple additive and multiplicative compositional functions. Other approaches, however, share with our proposal the use of syntax to drive modification of the distributional representation (Erk and Padó 2008; Thater, Dinu, and Pinkal 2009; Thater, Fürstenau, and Pinkal 2010, 2011).

Erk and Padó (2008) introduced a structured vector space model of word meaning that computes the meaning of a word in the context of another word via selectional preferences. This approach was shown to work well at ranking paraphrases taken from the SemEval-2007 lexical substitution task (McCarthy and Navigli 2007). In the Erk and Padó approach, the meaning of *ball* in the context of the phrase *catch ball* is computed by

combining the lexical vector for *ball* with the object preference vector of *catch* (i.e., things that can be *caught*). Although this approach is based on very similar intuitions to ours, it is in fact quite different. The lexical vector that is modified is not the co-occurrence vector, as in our model, but a vector of neighbors computed from co-occurrences. For example, the *lexical* vector for *catch* in the Erk and Padó approach might contain *throw*, *catch*, and *organize*. These neighbors of *catch* are then combined with verbs that have been seen with *ball* in the direct object relation using vector addition or component-wise multiplication. Thus, it is possible to carry out this approach with reference only to observed first-order grammatical dependency relationship. In their experiments, they used the "dependency-based" vector space of Padó and Lapata (2007) where target and context words are linked by a valid dependency path (i.e., not necessarily a single first-order grammatical relation). However, higher-order dependency paths were purely used to provide extra contexts for target words, than would be seen in a traditional first-order dependency model, during the computation of neighbor sets. Further, the Erk and Padó approach does not construct a representation of the phrase because this model is focused on lexical disambiguation rather than composition and it is not obvious how one would carry out further disambiguations within the context of a whole sentence.

More recently, Thater, Fürstenau, and Pinkal (2011) used a similar approach but considered a broader range of operations for combining two vectors where individual vector components are reweighted. Specifically, they found that reweighting vector components based on the distributional similarity score between words defining vector components and the observed context words led to improved performance at ranking paraphrases.

Thater, Fürstenau, and Pinkal (2010) noted that vectors of two syntactically related words typically have different syntactic environments, making it difficult to combine information in the respective vectors. They build on Thater, Dinu, and Pinkal (2009), where the meaning of argument nouns was modeled in terms of the predicates they co-occur with (referred to as a *first-order* vector) and the meaning of predicates in terms of *second-order* co-occurrence frequencies with other predicates. These predicate vectors can be obtained by adding argument vectors. For example, the verb *catch* will contain counts on the dimension for *kick* introduced by the direct-object *ball* and counts on the dimension for *contract* introduced by the direct-object *cold*. In other words, as in the Erk and Padó approach, the vector for a verb can be seen as a vector of similar verbs, thus making this notion of *second-order* dependency compatible with that used in work on word sense discrimination (Schütze 1998) rather than referring to second-order (or higher-order) grammatical dependencies as in this work. Contextualization can then be achieved by multiplication of a second-order predicate vector with a first-order argument vector because this selects the dimensions that are common to both. Thater, Fürstenau, and Pinkal (2010) presented a more general model where every word is modeled in terms of first-order and second-order co-occurrences and demonstrate high performance at ranking paraphrases.

7. Directions for Future Work

7.1 Representations

There are a number of apparent limitations of our approach that are simply a reflection of our decision to adopt dependency-based syntactic analysis.

First, surface disparities in syntactic structure (e.g., active versus passive tense formations, compound sentence structures) will disrupt sentence-level comparisons using a simple APT structure based on surface dependency relations, but this can be addressed, for example, by syntax-based pre-processing. The APT approach is agnostic in this regard.

Second, traditional dependency parsing does not distinguish between the order of modifiers. Hence the phrases *happiest blonde person* and *blonde happiest person* receive the same dependency representation and therefore also the same semantic representation. However, we believe that our approach is flexible enough to be able to accommodate a more sensitive grammar formalism that does allow for distinctions in modifier scope to be made if an application demands it. In future work we intend to look at other grammar formalisms, including CCG (Steedman 2000).

By proposing a count-based method for composition we are bucking the growing trend of working with prediction-based word embeddings. Although there has been initial evidence (Baroni, Dinu, and Kruszewski 2014) that prediction-based methods are superior to count-based methods at the lexeme level (e.g., for synonym detection and concept categorization), it has also been shown (Levy and Goldberg 2014) that the skip-gram model with negative sampling as introduced in Mikolov et al. (2013a) is equivalent to implicit factorization of the PPMI matrix. Levy, Goldberg, and Dagan (2015) also demonstrated how traditional count-based methods could be improved by transferring hyperparameters used by the prediction-based methods (such as context distribution smoothing and negative sampling). This led to the count-based methods outperforming the prediction-based methods on a number of word similarity tasks. A next step for us is to take the lessons learned from work on word embeddings and find a way to produce lower dimensionality APT representations without destroying the necessary structure that drives composition. The advantages of this from a computational point of view are obvious. It remains to be seen what effect the improved generalization also promised by dimensionality reduction will have on composition via APTs.

By considering examples, we have seen that composition of APTs using both union and intersection can lead to nearest neighbors that are clearly disambiguating. On benchmark phrase-based composition tasks, the performance of union in APT composition is close to or equal to the state of the art on those tasks. However, we believe that the performance of intersection in APT composition is currently limited by the impoverished nature of word representations based directly on corpus statistics. Even given a very large corpus, there are always many plausible co-occurrences that have not been observed. One possible solution, which we explore elsewhere, is to smooth the word representations using their distributional neighbors before applying an intersective composition operation.

7.2 Applications

In Section 5.2, we demonstrated the potential for using APTs to carry out word sense disambiguation / induction. Uncontextualized, elementary APTs typically contain a corpus-determined mixture of co-occurrences referencing different usages. The APT generated by a dependency tree, however, provides contextualized lexeme representations where the weights have been adjusted by the influence of the contextual lexemes so that the co-occurrences relating to the *correct* usage have been appropriately up-weighted, and the co-occurrences found in other circumstances down-weighted. In

other words, APT structures automatically perform word sense induction on lexeme-level representations and this is demonstrable through the lexeme similarity measure. For example, we observed that the contextualized lexeme representation of *body* in the APT constructed by embedding it in the phrase *human body* had a relatively high similarity to the uncontextualized representation of *brain* and a relatively low similarity to *council*, whereas the equivalent lexeme representation for *body* embedded in the APT constructed for the phrase *legislative body* showed the reverse pattern.

One common criticism of distributional thesauruses is that they conflate different semantic relations into a single notion of similarity. For example, when comparing representations based on grammatical dependency relations, the most similar word to an adjective such as *hot* will usually be found to be its antonym *cold*. This is because *hot* and *cold* are both used to modify many of the same nouns. However, if, as in the APT framework, the representation of *cold* includes not only the direct dependents of *cold*, but also the indirect dependents (e.g., verbs that co-occur with *cold things*), it is possible that more differences between its representation and that of *hot* might be found. One would imagine that the things that are done to *hot things* are more different to the things that are done to *cold things* than they are to the things that are done to *very warm things*. Further, the examples in Section 5.2 raise the possibility that different composition operations might be used to distinguish different semantic relations including hypernyms, hyponyms, and co-hyponyms. For example, $\bigsqcup_{\text{UNI}}$ tends to lead to more general neighbors (e.g., hypernyms) and $\bigsqcup_{\text{INT}}$ tends to lead to more specific neighbors (e.g., hyponyms).

Phrase-level or sentence-level plausibility measures offer the prospect of a continuous measure of the *appropriateness / plausibility* of a complete phrase or sentence, based on a combination of semantic and syntactic dependency relations. APTs offer a way to measure the plausibility of a lexeme when embedded in a dependency tree, suggesting that APTs may be successfully used in tackling sentence completion tasks, such as the Microsoft Research Sentence Completion Challenge (Zweig and Burges 2012). Here, the objective is to identify the word that will fill out a partially completed sentence in the best possible way. For example, is *flurried* or *profitable* the best completion of the sentence below:

> "Presently he emerged looking even more *[flurried / profitable]* than before."

We can compose the APTs for the partially completed sentence. Comparing the result with the elementary APTs for each of the candidates should provide a good, direct measurement of which candidate is more plausible. An improved language model has implications for parsing, speech recognition, and machine translation.

A central goal of compositional distributional semantics is to create a data structure that represents an entire phrase or sentence. The composed APT for a dependency tree provides such a structure, but leaves open the question as to how this structure might be exploited for phrase-level or sentence-level semantic comparison.

The first point to be made is that, unusually, we have available not only a representation of the whole dependency tree but also contextualized (vector) representations for the lexemes in the dependency tree. This makes available to us any analytical technique that requires separate analysis of lexical components of the phrase or sentence. However, this leads to the problem of how to *read* the structure at the global phrase/sentence-level.

For similarity measures, one straightforward option would be to create a vector from the APT anchored at the head of the phrase or sentence being considered. Thus, the phrasal vector for *a red rose* would be created taking the node containing *rose* as the

anchor. In other words, the vector representation of the phrase *a red rose* will be the same as the contextualized representation of *rose*. Similarly, the vector representation for the sentence *he took the dog for a walk* will be the same as the contextualized representation of the verb *took*.

Such a representation provides a continuous model of similarity (and meaning) at the phrasal-level and/or sentence-level. We anticipate that vector comparisons of phrase- or sentence-level vectors produced in this manner will provide some coherent numerical measure of distributional similarity. This approach should be useful for paraphrase recognition tasks. For example, in order to identify good candidate paraphrases for questions in a question-answering task, Berant and Liang (2014) use a paraphrase model based on adding word embeddings constructed using the CBOW model of Mikolov et al. (2013). Although the authors achieve state of the art using a mixture of methods, a paraphrase model based on the addition of vectors of untyped co-occurrences alone cannot distinguish meanings where syntax is important. For example, the sentences *Oswald shot Kennedy* and *Kennedy shot Oswald* would have the same representations. On the other hand, APT composition is syntax-driven and will provide a representation of each sentence that is sensitive to lexical meaning and syntax.

Another advantage of using APT composition in paraphrase recognition, over some other syntax-driven proposals, is that the same structure is used to represent words, phrases, and sentences. Provided that the head node is of the same type of speech, words and phrases of different lengths can easily be compared within our model. An adjective–noun compound such as *male sibling* is directly comparable with the single noun *brother*. Further, there is no need for there to be high similarity between aligned components of phrases or sentences. For example, the phrase *female scholar* can be expected to have a high similarity with the phrase *educated woman*, in terms at least of their external contexts.

8. Conclusions

This article presents a new theory of compositional distributional semantics. It uses a single structure, the APT, which can represent the distributional semantics of lexemes, phrases, and even sentences. By retaining higher-order grammatical structure in the representations of lexemes, composition captures mutual disambiguation and mutual generalization of constituents. APTs allow lexemes and phrases to be compared in isolation or in context. Further, we have demonstrated how one instantiation of this theory can achieve results that are very competitive with state-of-the-art results on benchmark phrase-based composition tasks.

As we have discussed, APTs have a wide range of potential applications including word sense induction, word sense disambiguation, parse reranking, dependency parsing and language modeling more generally, and also paraphrase recognition. Further work is required to gain an understanding of which instantiations of the theory are suited to each of these applications.

Acknowledgments
This work was funded by the UK EPSRC project EP/IO37458/1 "*A Unified Model of Compositional and Distributional Compositional Semantics: Theory and Applications*." We would like to thank all members of the DISCO project team. Particular thanks to Miroslav Batchkarov, Stephen Clark, Daoud Clarke, Roland Davis, Bill Keller, Tamara Polajnar, Laura Rimell, Mehrnoosh Sadrzadeh, David Sheldrick, and Andreas Vlachos. We would also like to thank the anonymous reviewers for their helpful comments.

References

Baroni, Marco, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247, Baltimore, MD.

Baroni, Marco and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4): 673–721.

Baroni, Marco and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1183–1193, Cambridge, MA.

Berant, Jonathan and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425, Baltimore, MD.

Blacoe, William and Mirella Lapata. 2012. A comparison of vector-based representations for semantic composition. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 546–556, Jeju Island.

Bruni, Elia, Nam Khanh Tran, and Marco Baroni. 2014. Multimodal distributional semantics. *Journal of Artificial Intelligence Research*, 49:1–47.

Clarke, Daoud. 2007. *Context-Theoretic Semantics for Natural Language: an Algebraic Framework*. Ph.D. thesis, Department of Informatics, University of Sussex.

Clarke, Daoud. 2012. A context-theoretic framework for compositionality in distributional semantics. *Computational Linguistics*, 38(1):41–71.

Coecke, Bob, Mehrnoosh Sadrzadeh, and Stephen Clark. 2011. Mathematical foundations for a compositional distributed model of meaning. *Linguistic Analysis*, 36(1-4):345–384.

Collobert, Ronan and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 160–167, New York, NY.

Curran, James. 2004. *From Distributional to Semantic Similarity*. Ph.D. thesis, University of Edinburgh.

Dagan, Ido, Fernando Pereira, and Lillian Lee. 1994. Similarity-based estimation of word cooccurrence probabilities. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 272–278, Las Cruces, NM.

Dinu, Georgiana, Nghia The Pham, and Marco Baroni. 2013. General estimation and evaluation of compositional distributional semantic models. In *Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality*, pages 50–58, Sofia.

Dinu, Georgiana and Stefan Thater. 2012. Saarland: Vector-based models of semantic textual similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics*, pages 603–607, Montreal.

Erk, Katrin. 2012. Vector space models of word meaning and phrase meaning: A survey. *Language and Linguistics Compass*, 6(10):635–653.

Erk, Katrin and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 897–906, Honolulu, HI.

Ferraresi, Adriano, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. 2008. Introducing and evaluating UKWAC, a very large Web-derived corpus of English. In *Proceedings of the WAC4 Workshop at LREC*, pages 47–54, Marrakech.

Finkelstein, Lev, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th International Conference on World Wide Web*, pages 406–414, New York, NY.

Garrette, Dan, Katrin Erk, and Raymond Mooney. 2011. Integrating logical representations with probabilistic information using Markov logic. In *Proceedings of the Ninth International Conference on Computational Semantics*, pages 105–114, Stroudsburg, PA.

Grefenstette, Edward, Georgiana Dinu, Yao-Zhong Zhang, Mehrnoosh Sadrzadeh, and Marco Baroni. 2013. Multi-step regression learning for compositional distributional semantics. In *Proceedings of the Tenth International Conference on Computational Semantics*, pages 131–142, East Stroudsberg, PA.

Guevara, Emiliano. 2010. A regression model of adjective-noun compositionality in distributional semantics. In *Proceedings of the ACL Geometrical Models of Natural Language Semantics Workshop*, pages 33–37, Uppsala.

Hermann, Karl Moritz and Phil Blunsom. 2013. The role of syntax in vector space models of compositional semantics. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 894–904, Sofia.

Hill, Felix, Roi Reichart, and Anna Korhonen. 2015. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Kiela, Douwe and Stephen Clark. 2014. A systematic study of semantic vector space model parameters. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*, pages 21–30, Gothenburg.

Lambek, J. 1999. Type grammar revisited. In Alain Lecomte, François Lamarche, and Guy Perrier, editors, *Logical Aspects of Computational Linguistics*, volume 1582 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, pages 1–27.

Lee, Lillian. 1999. Measures of distributional similarity. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 25–32, College Park, MD.

Levy, Omer and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*. Curran Associates, Inc., pages 2177–2185.

Levy, Omer, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Lewis, Mike and Mark Steedman. 2013. Combined distributional and logical semantics. In *Transactions of the Association for Computational Linguistics*, 1:179–192.

Lin, Dekang. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pages 768–774, Montreal.

Lund, Kevin and Curt Burgess. 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, & Computers*, 28(2):203–208.

McCarthy, Diana and Robert Navigli. 2007. Semeval-2007 task 10: English lexical substitution task. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007)*, pages 48–53, Prague.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc., pages 3111–3119.

Mikolov, Tomas, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, GA.

Mitchell, Jeff and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technology Conference*, pages 236–244, Columbus, OH.

Mitchell, Jeff and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429.

Montague, Richard. 1970. English as a formal language. In Bruno Visentini, editor, *Linguaggi nella società e nella tecnica*, pages 189–223.

Nivre, Joakim. 2004. Incrementality in deterministic dependency parsing. In *Proceedings of the ACL Workshop on Incremental Parsing*, pages 50–57, Boston, MA.

Padó, Sebastian and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199.

Pennington, Jeffrey, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In

Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pages 1532–1543, Doha.

Schütze, Hinrich. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123.

Socher, Richard, Eric H. Huang, Jeffrey Pennington, Christopher D. Manning, and Andrew Y. Ng. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, 24:801–809.

Socher, Richard, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211, Jeju Island.

Socher, Richard, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 151–161, Edinburgh.

Socher, Richard, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, WA.

Steedman, Mark. 2000. *The Syntactic Process*. MIT Press.

Thater, Stefan, Georgiana Dinu, and Manfred Pinkal. 2009. Ranking paraphrases in context. In *Proceedings of the 2009 ACL Workshop on Applied Textual Inference*, pages 44–47, Suntec.

Thater, Stefan, Hagen Fürstenau, and Manfred Pinkal. 2010. Contextualizing semantic representations using syntactically enriched vector models. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 948–957, Uppsala.

Thater, Stefan, Hagen Fürstenau, and Manfred Pinkal. 2011. Word meaning in context: A simple and effective vector model. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, pages 1134–1143, Chiang Mai.

Turney, Peter D. 2012. Domain and function: A dual-space model of semantic relations and compositions. *Journal of Artificial Intelligence Research*, 44(1):533–585.

Turney, Peter D. and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188.

Van de Cruys, Tim, Thierry Poibeau, and Anna Korhonen. 2011. Latent vector weighting for word meaning in context. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1012–1022, Edinburgh.

Weeds, Julie. 2003. *Measures and Applications of Lexical Distributional Similarity*. Ph.D. thesis, Department of Informatics, University of Sussex.

Weeds, Julie and David Weir. 2005. Co-occurrence retrieval: A flexible framework for distributional similarity. *Computational Linguistics*, 31(4):439–476.

Weeds, Julie, David Weir, and Diana McCarthy. 2004. Characterising measures of lexical distributional similarity. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 1015–1021, Geneva.

Weeds, Julie, David Weir, and Jeremy Reffin. 2014. Distributional composition using higher-order dependency vectors. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*, pages 11–20, Gothenburg.

Wilson, Benjamin. 2015. The unknown perils of mining wikipedia. `https://blog.lateral.io/2015/06/the-unknown-perils-of-mining-wikipedia/`.

Zweig, Geoffrey and Chris J. C. Burges. 2012. A challenge set for advancing language modeling. In *Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Replace the N-gram Model? On the Future of Language Modeling for HLT*, pages 29–36, Montreal, CA.